ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO TÉCNICO EN INFORMÁTICA DE GESTIÓN

Título del proyecto:

MONITORIAZACIÓN DE PARCELAS AGRARIAS EN ENTORNO WEB.

Alumno: Amaiur Arratibel Arrula

Tutores: José Javier Astrain Escola
Jesús Villadangos Alonso

Pamplona, 9 de Septiembre de 2013

Índice de contenidos

Índice de ilustraciones

Capítulo 1 Introducción

La Fundación Conama, una organización española independiente y sin ánimo de lucro, que promueve el intercambio de conocimientos en pos del desarrollo sostenible, suele organizar anualmente congresos internacionales. Estos puntos de encuentro tienen el objetivo de escuchar a diferentes interlocutores del sector ambiental (profesionales, académicos, empresas, ecologistas…), y promover la colaboración entre ellos para avanzar en un desarrollo sostenible. Las siguientes líneas recogidas de su último congreso, sintetizan perfectamente la situación actual del sector agrario y el porqué del desarrollo de este proyecto:

"El uso abusivo e irracional de los recursos naturales en la agricultura española (ineficiente utilización de bienes agrícolas, sobreexplotación del suelo y de los recursos hídricos) ha conducido a que un tercio de la superficie cultivable se encuentre con niveles graves de degradación. Además, el sector agrícola arrastra en los últimos años una problemática agravada por el momento actual de recesión económica: progresivo envejecimiento y emigración de la población activa hacia otros países económicos.

La dimensión media de las explotaciones españolas – 23 Ha de Superficie Agraria Útil – sigue encontrándose entre las más pequeñas de la UE, junto con las de Austria (19,1), Portugal (11,4), Italia (7,4) y Grecia (4,8), indicador de debilidad estructural que limita la competitividad del sector en su conjunto. En los últimos años se aprecia una desaceleración del proceso de ajuste estructural, lo que da a entender que el aumento del rendimiento de las explotaciones no depende tanto del aumento de la dimensión como de su desplazamiento hacia orientaciones con mayor productividad.

En regiones en las que predomina el minifundismo, existen además problemas derivados de sus inadecuadas estructuras de producción: excesiva parcelación, deficiencias de la formación profesional del personal, reducida implantación de las estructuras cooperativas, desorganización e individualismo en los canales de distribución y comercialización, marcada dispersión y atomización de las explotaciones y de los métodos de trabajo y de las tecnologías utilizadas.

El reto está claro: es preciso potenciar la producción agrícola, pero con parámetros de sostenibilidad. La agricultura de precisión es un tipo de agricultura más respetuosa con el medioambiente, ya que realiza buenas aplicaciones de los bienes disponibles (fertilizantes, fitosanitarios, agua) según las necesidades del cultivo, lo que repercute también en un ahorro de costes, aumento de la calidad y productividad. La telemetría, la localización por satélite y los sistemas de información geográfica (GIS), son la base sobre la cual se sustenta este tipo de agricultura. Debido a que la rentabilidad de la agricultura de precisión es dependiente de la localización, importar la tecnología de otros países o regiones no es tan efectivo como en otros casos, lo que hace necesario desarrollar sistemas escalables pero que a la vez tengan en cuenta condiciones locales.

La telemetría basada en redes de sensores inalámbricas (tecnología WSN, Wireless Sensor Networks) se plantea como herramienta facilitadora de técnicas de agricultura de precisión, haciendo posible que los datos de evolución de parámetros críticos de la explotación estén accesibles por el gestor del cultivo en todo momento (tiempo real), lugar (remotamente a través de Internet) y a través de cualquier dispositivo (PDA, móvil, PC…), todo ello con un mínimo impacto, bajo consumo y bajo coste, de manera que pueda tomar decisiones a tiempo. Estas técnicas permiten la mejora del rendimiento y eficiencia de cultivos en explotaciones agrarias típicamente minifundistas, contribuyendo así al desarrollo de una agricultura sostenible."

1.1. Descripción y objetivos

El problema a resolver trata de implementar un entorno web desde el cual tener monitorizado un conjunto de parcelas agrarias, y así poder visualizar y controlar los datos recogidos en las mismas.

El objetivo principal es dotar al usuario de control absoluto sobre sus parcelas. Esto es posible ya que la web está ínter-relacionada con los sensores colocados en las parcelas, los cuales recogerán datos característicos del terreno (humedad, mm^3 de lluvia caída, radiación solar...), siendo posible en todo momento y lugar el acceso a los datos. El usuario evitará tener que ir a tomar mediciones de forma manual y trasladarse entre sus distintos terrenos, además de tener que registrar él mismo los datos recogidos.

La principal utilidad de este proyecto es poder explotar al máximo las parcelas, debido a la constante recogida de medidas que al instante se pueden visualizar. Se trata de una explotación "estudiada" y controlada de la forma más certera.

El proyecto debe ser íntegro y duradero, para que actúe de manera automática e independiente al usuario, sin tener que preocuparse por su mantenimiento y continuo funcionamiento. De esta manera el usuario podrá disfrutar de las múltiples utilidades y ventajas de la aplicación.

Además, esta aplicación debe estar sincronizada, para que así estén disponibles los datos más recientes en el tiempo. Si a esto le añadimos datos externos reales, ya se dispondrían de todo tipo de datos y variantes.

El punto más importante es la usabilidad final de la aplicación, sobre todo, teniendo en cuenta que se trata de un entorno web, en el cual lo primordial es una buena y sencilla interacción entre el usuario y el software. Todas las funcionalidades de la aplicación deben poder ser intuidas por el usuario, consiguiendo una utilización de la misma del cien por ciento.

Resumiendo, la aplicación debe resolver el problema de monitorizar parcelas, para que así el propietario pueda tener controlado en todo momento y lugar las características agrarias de dichas parcelas. Con esto se optimiza la producción de los

terrenos, y se consigue reducir la medición manual, con sus correspondientes traslados.

1.2. Estado del arte

1.2.1. Recursos exteriores

Hoy en día toda explotación agraria se realiza a la par de un estudio, comenzando con un estudio previo de la zona de cultivo, situación, temperatura, terreno... y a continuación se lleva a cabo el cultivo y su posterior seguimiento. Este seguimiento se realiza en sucesivas campañas diferenciadas en las cuales se sigue un calendario realizado de antemano, lo que conlleva una total dependencia del cultivo. Dicho estudio concluye con la creación de una gran y potente base de datos, que será utilizada para caracterizar el suelo en una línea temporal.

Para realizar un buen estudio hay que apoyarse en fuentes externas y fiables, las cuales proporcionen datos de máxima exactitud. Actualmente existen varias agencias de meteorología que ponen al servicio del usuario de a pie tanto datos de predicciones, como completas bases de datos con lo recogido en tiempos pasados. A nivel estatal el referente es la Agencia Estatal de Meteorología (AEMET) y a nivel foral el servicio de Meteorología y Climatología de Navarra, que se complementa de AEMET. A continuación, vamos a profundizar un poco más en ambas agencias meteorológicas, con información de primera mano proporcionada por ellas vía email.

AEMET

La Agencia Estatal de Meteorología fue creada por Real Decreto en 2008 sustituyendo al antiguo Instituto Nacional de Meteorología. Se encuentra adscrita al Ministerio de Agricultura, Alimentación y Medio Ambiente a través de la Secretaría de Estado de Medio Ambiente. Además de ejercer la autoridad meteorológica, representa al Estado ante los organismos internacionales de meteorología. Con sede en Madrid, posee varios Centros Meteorológicos repartidos por diferentes Comunidades autónomas, entre las que está Navarra. También cuenta con Oficinas Meteorológicas en aeropuertos y diversos observatorios.

El objeto de AEMET es el desarrollo, implantación, y prestación de los servicios meteorológicos de competencia del Estado. También se encarga de dar apoyo a actividades tanto públicas como privadas, contribuyendo a la seguridad y bienestar de la sociedad.

Las competencias y funciones de la Agencia Estatal parten del suministro y difusión de las informaciones meteorológicas y predicciones. A partir de ahí, amplían sus competencias en diferentes campos:

- Da respuesta al interés despertado por la ciudadanía, alertados con avisos de los fenómenos meteorológicos que puedan afectar a su integridad física y bienes materiales.
- Aporta información a los organismos de protección civil (ejército, navegación aérea, marítima, defensa nacional...), contribuyendo en la su seguridad y eficiencia.
- Prestación de servicio a las políticas medioambientales de asesoramiento científico en asuntos relacionados con la variabilidad y el cambio climático.
- Mantenimiento de una vigilancia continua, eficaz y sostenible de las condiciones meteorológicas, climáticas y de la estructura y composición física y química de la atmósfera sobre el territorio nacional.
- Mantenimiento y permanente actualización del registro histórico de datos meteorológicos y climatológicos.
- Realización de estudios e investigaciones en los campos de las ciencias atmosféricas, para el progreso en el conocimiento del tiempo y clima, a través de lo obtenido en las diferentes redes de observación, con la adecuada adaptación a la tecnología.
- Elaboración y actualización de los escenarios de cambio climático.
- Prestación de servicios meteorológicos a la carta (servicios de pago).

Además de los servicios convencionales de observación meteorológica y climatológica, los servicios más interesantes y útiles de AEMET son

- Imágenes de satélite y de radar
- Predicciones
- Mapas de rayos
- Radiación ultravioleta
- Meteorología marítima
- Información aeronáutica

Para llevar a cabo todos estos servicios y actividades, la Agencia cuenta con los siguientes medios:

- *Personal*: una plantilla de 1400 personas, de las cuales el 69% se encuentra en las Delegaciones Territoriales y el 31% restante desarrolla su actividad en la Sede Central.
- *Presupuesto anual*: 122.796.520 €
- *Potencia de cálculo*: ordenador Cray X1E con una potencia pico de cálculo de 2380 Gflops y con una potencia máxima calculada de más de 1800 Gflops. La velocidad de cálculo sostenida por este ordenador, para el modelo HIRLAM/ AEMET, modelo numérico de predicción de tiempo, es cercana a los 700 Gflops*.
- *Datos de satélites meteorológicos*: Recepción y proceso de imágenes y datos de los satélites geoestacionarios METEOSAT y GOES-este y de los polares TIROS-NOAA y METOP.
- *Red de observación*:
  90 observatorios con personal propio de la Agencia.
  700 estaciones automáticas de observación.
  7 estaciones de radiosondeo en tierra, 1 en el buque "Esperanza del Mar" y 2 en las oficinas meteorológicas móviles de Defensa.
  Red de 15 radares meteorológicos con capacidad doppler.
  Red de detección de rayos con 15 equipos detectores en la Península y 5 en las Islas Canarias.
  Red de 4.500 estaciones pluviométricas y termopluviométricas atendidas por colaboradores altruistas.

*FLOPS, floating point operations per second, es una medida del rendimiento de una computadora, contabilizando el número de operaciones de coma flotante por segundo, especialmente en cálculos científicos que requieren un gran uso de operaciones de coma flotante.

Meteorología y climatología de Navarra

Las estaciones meteorológicas manuales en Navarra comenzaron a instalarse en 1880 (estación de Pamplona). Entre 1900 y 2000 el Instituto Nacional de Meteorología, precursor de la AEMET, dio forma a la actual red de estaciones meteorológicas manuales. Actualmente funcionan 87 en Navarra.

Las estaciones automáticas en Navarra comenzaron a instalarse en el año 1991. Actualmente el Gobierno de Navarra tiene instaladas 26 estaciones de este tipo, de las que 18 envían datos en tiempo real y el resto envían los datos tres veces al día.

En el año 1998 se creó la Base de Datos unificada de meteorología de Navarra, con todos los datos de las estaciones existentes tanto manuales como automáticas, del Gobierno de Navarra, AEMET y otras redes (INTIA y Ministerio).

La primera versión de la web de meteorología se creó en el año 2000. En un principio tan sólo mostraba los datos recogidos en las estaciones, pero fue evolucionando incluyendo datos meteorológicos en tiempo real, predicciones, gráficos y más contenidos.

Este servicio es gestionado por el Negociado de Suelos y Climatología, perteneciente al departamento de Desarrollo Rural, Medio Ambiente y Administración Local, con la colaboración de la Agencia Estatal de Meteorología (AEMET) y de la empresa pública TRAGSATEC.

El departamento de Desarrollo Rural, Medio Ambiente y Administración Local se encuentra en la c/ González Tablas 9, 31005 Pamplona-Iruña.

El objetivo es dar servicio tanto a la administración como a la sociedad en general. Se envían datos meteorológicos a las personas o entidades que los soliciten, además gran parte de los datos se pueden descargar directamente de la web. La información meteorológica (datos actuales de las estaciones meteorológicas, datos históricos, medias) es demandada en muchos campos:

- AEMET (generación de predicciones meteorológicas), Obras públicas, Medio Ambiente (prevención de incendios), empresas de energías renovables, Protección Civil (prevención de inundaciones), Medios de comunicación, estudios hidrológicos, agricultura (necesidades de riego, control de plagas), Sanidad (prevención de olas de calor), deportes aéreos, montañismo, estudios relacionados con el cambio climático. También los usuarios particulares demandan información para estudios, proyectos de fin de carrera, ocio, interés en la climatología.

Trabajan en este servicio personas del Negociado de Suelos y Climatología, de la empresa TRAGSATEC, y 87 observadores de las estaciones manuales.

Las agencias de meteorología anteriormente vistas sirven de gran ayuda en el día a día de multitud de empresas de explotación agraria, para poder anticiparse a las predicciones y poder realizar comparaciones entre distintos intervalos de tiempo. Además de esto es necesaria una monitorización medioambiental para conseguir el máximo rendimiento posible de las parcelas. ¿Qué es y en qué consiste la monitorización ambiental? ¿En qué se diferencia de la agricultura tradicional?

Monitorización medioambiental

Monitorización: *"supervisión necesaria que nos asegura eficacia en el resultado final, nos aseguramos que nuestro proyecto está encaminado adecuada y eficazmente hacia un resultado final, evitando las posibles desviaciones que pudieran presentarse."* [Marcos Martínez Peiro, BALMART 2010]

El objetivo de la monitorización medioambiental es principalmente corregir el procedimiento antes de llegar al resultado final. Esto implica una optimización de la producción agrícola, con lo que se evita el exceso de recursos (agua, abonos...) y se incrementa la calidad y la cantidad de la producción final.

Este tipo de agricultura se la conoce como **Agricultura de Precisión**, dicha agricultura se basa en el estudio y manejo de microclimas para mejorar el rendimiento y la cantidad de la cosecha. Además, permite aplicar métodos y técnicas específicas para cada cultivo y área geográfica.

¿Qué supone para el agricultor la agricultura de precisión?

Claramente un beneficio importante. Primeramente, el agricultor notará anualmente un ahorro económico por la optimización de recursos, personal... lo que conlleva un incremento de los beneficios brutos de la empresa. Otro punto a destacar es el uso de tratamientos químicos de forma precisa, sólo se utilizan cuándo y dónde son necesarios, consiguiendo un producto más puro y natural. Con este control, se regula el uso del agua para el riego, lo que normalmente lleva a una mala gestión con problemas en tiempos de sequías y derroche en tiempos de abundancia, ahora será una toma de decisiones tácticas respecto al momento y la duración del riego a lo largo del ciclo de cultivo. En cuanto a beneficio temporal, se consigue saber exactamente cuándo interesa cosechar, regar, fertilizar... en función de las condiciones meteorológicas y de cultivo. Finalmente, el usuario tendrá un control regulado de las posibles heladas en sus cosechas, así como la degradación de los suelos.

Los puntos que principalmente controla la agricultura de precisión son los siguientes:

- Condiciones meteorológicas: aquí se controlan puntos tan importantes como la temperatura ambiente, la humedad ambiente, la cantidad de lluvia, viento, rocío...
- Condiciones del terreno: utilizado para realizar estudios relacionados con la compactación del terreno, la temperatura y humedad de la tierra a distintos niveles, el pH, la conductividad…
- Condiciones ambientales: control de la luminosidad, radiación solar, niveles de CO_2, posibles plagas…
- Condiciones del cultivo: control más exhaustivo del producto, estudiando los niveles de clorofila, diámetros, grosores, aspecto, calibre…

Con el estudio realizado correspondiente, se llega al nivel de control casi total, llevando al detalle las épocas de reforestación, fertilización, salinidad, heladas, cosechas y sembrado.

Los actores que se encargan del seguimiento y estudio de esta agricultura son principalmente los que siguen:

- Propietarios agricultores
- Laboratorio de análisis
- Gabinete de ingeniería
- Técnicos de campo de la explotación

Dichos actores se desenvuelven en un aprendizaje adaptativo, en el cual comienzan con un estudio de necesidades, seguido toman muestras con los sensores y por último se realiza el análisis de datos; tras esto, se comienza el ciclo de estudiar las necesidades comparando con lo obtenido en el aprendizaje anterior. La siguiente ilustración, realizada por una empresa de la que luego hablaremos, recoge claramente este aprendizaje.

Ilustración 1: Ciclo de monitorización medioambiental

Comparando el método de monitorización con el que se ha usado hasta ahora, el **procedimiento de medida clásico**, nos damos cuenta de que este último es mucho más costoso. Este método que se ha llevado realizando durante años comienza con la toma de muestras en el campo o mediante una estación meteorológica colocada en el mismo. Seguido es necesario que el agricultor realice visitas a sus terrenos, ya sea de forma diaria, semanal, o mensual, para así ir controlando la evolución. Allí se recogen los datos de forma manual, que se transcribirá en documentos para mantenerlos almacenados y poder disponer de ellos. Estos datos no son entendibles por el agricultor, y se necesita disponer de unos expertos que los analicen meticulosamente y realicen los estudios necesarios. Una vez concluido el proceso de análisis de los datos, ya se puede tomar decisiones sobre la explotación para mejorar el rendimiento en un futuro próximo.

Los defectos de la medición clásica resaltan claramente. Primero destacar el exceso de tiempo que se necesita para tomar las decisiones en el proceso de análisis de los terrenos. A esto le tenemos que sumar todo el coste monetario que suponen la multitud de acciones a realizar, desde traslados en vehículos al lugar de medición, hasta el coste de los equipos técnicos y tecnológicos necesarios.

La consecuencia de esta agricultura ha sido que la monitorización sea de escaso uso, centrándolo mayoritariamente en estudios de índole científico, universidades, centros de investigación. A continuación, una breve pero concisa imagen recoge estos pasos de la agricultura clásica.

Ilustración 2: Procedimiento de medida clásico

A diferencia de la agricultura clásica, en la agricultura de precisión se centra gran parte la labor en la tecnología. Continuamente, el responsable o propietario hará uso de las tecnologías de la información y la comunicación, como son todos los dispositivos relacionados con los pc's y los *smartphones*, siempre conectados y comunicados con la red. Está interacción se da con los medidores distribuidos geográficamente que tomarán las muestras, que más tarde quedarán almacenados en bases de datos informáticas, que aportan mucha más fiabilidad que papeles físicos que se pueden dañar. Seguidamente los datos recogidos se pueden visualizar automáticamente mediante informes realizados en tiempo real con un coste temporal minúsculo, y dicho acceso a la información se podrá hacer desde cualquier parte del mundo sin tener que acceder personalmente a los datos.

Toda esta tecnología reporta infinitud de ahorro y beneficio, consiguiendo primordialmente el aumento de las ganancias de la explotación. Los costes de implantación se ven amortizados ya que se evita el coste humano y de transporte, que con el tiempo se sigue acumulando y es cuantioso. Esta implantación de tecnología hace posible que el tiempo de respuesta entre el usuario y la información sea instantáneo, consiguiendo una reducción de tiempo más que considerable. Además que el hardware y el software necesarios para la monitorización son de bajo coste, siendo accesible una Agronomía Profesional a los distintos niveles adquisitivos.

En la administración y monitorización de regadíos se ha invertido tanto a nivel autonómico como a nivel estatal, fomentando el uso y la inversión de redes de control de regadíos. A esto, se le añade la inversión tecnológica adecuada al regadío, consiguiendo que el agua llegue a todos los consumidores con control remoto del bombeo, la distribución y los contadores. Esta forma de regular el riego ayuda al control del pago, ya que la apertura y cierre del riego esta automatizado. Las inversiones en esta área sólo se han apoyado desde el punto de vista científico, aunque actualmente se está apoyando en distintas comunidades.

Consejos a la hora de monitorizar:

- Es conveniente disponer de un sistema multisensorial, para supervisar varios parámetros de interés como las características del suelo y los factores ambientales.
- Es conveniente disponer de datos distribuidos en el terreno y no localizados solamente en un punto.
- Las redes inalámbricas de sensores (WSN) están iniciando su presencia en el espacio agrícola y del medio ambiente.

Sistemas de Información Medioambiental

Los sistemas de información (SI) fueron diseñados para satisfacer las necesidades de gestión de información que existen en cualquier organización. Son sistemas que toman datos de la organización y del exterior para, una vez manipulados y analizados convenientemente, producir nueva información que apoye la toma de decisiones. En la siguiente ilustración podemos observar el diagrama simplificado de un SI, caracterizado por la presencia de lazos de realimentación que permiten volver a una fase anterior si así requieren los resultados obtenidos en etapas sucesivas.

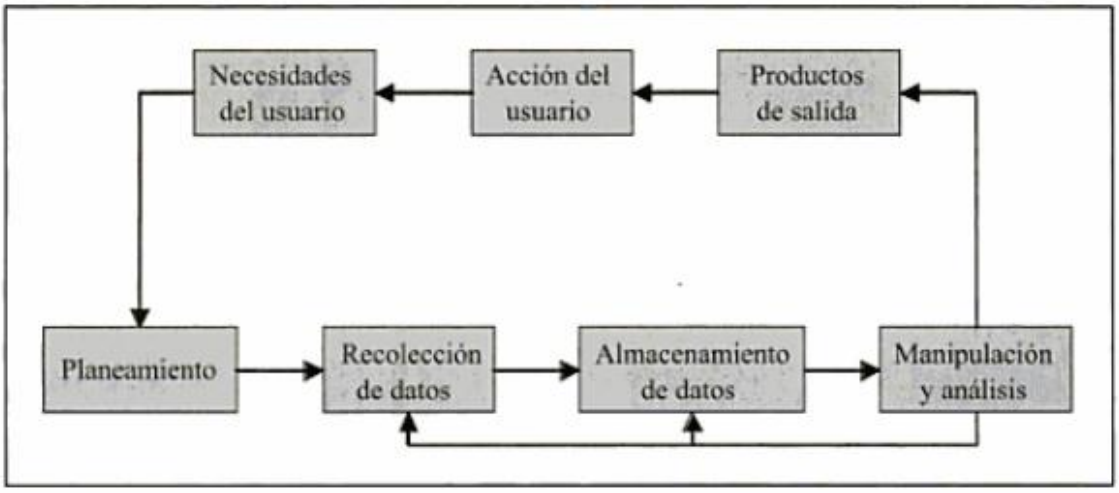


Ilustración 3: Diagrama simplificado de un SI
Fuente: "*Sistemas de Información Medioambiental*" [2005]

Desde la anterior perspectiva, un mapa es un sistema de información sencillo, ya que consiste en una colección de datos que son examinados para extraer información relevante en la toma de decisión. Por extensión, un SIG puede ser visto como cualquier "sistema de información diseñado para trabajar con datos referenciados a través de coordenadas espaciales o geográficas. En otras palabras, un SIG es una base de datos con capacidades específicas para gestionar datos referenciados espacialmente, más un conjunto de operaciones para trabajar con ellos" [STAR Y ESTES, 1990].

Otros autores incluyen una definición más exhaustiva, según la cual un SIG es todo aquel "sistema de manejo de información capaz de:

- Coleccionar, almacenar y extraer información en función de su localización espacial.
- Identificar localizaciones dentro de un área de interés que cumplen unos criterios específicos.
- Explorar las relaciones entre conjuntos de datos dentro de ese entorno.
- Analizar espacialmente los datos relacionados como ayuda a la toma de decisiones acerca del entorno.

- Facilitar la selección y la transmisión de datos a modelos analíticos específicos capaces de evaluar el impacto inherente a las alternativas elegidas.
- Representar el entorno seleccionado, tanto gráfica como numéricamente, antes y después del análisis."

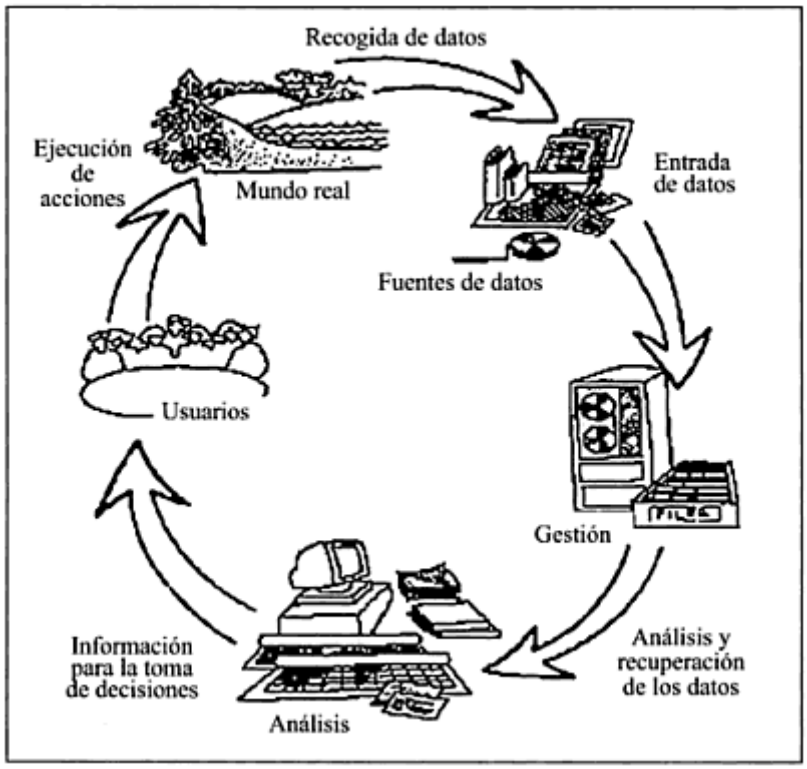


Ilustración 4: Procesado de la información geográfica
Fuente: "*Sistemas de Información Medioambiental*" [2005]

Por otro lado, el mundo de los SIG ha asistido en los últimos años a una explosión de aplicaciones destinadas a mostrar y editar cartografía en entornos web como Google Maps, Bing Maps u OpenStreetMap entre otros. Estos sitios web dan al público acceso a enormes cantidades de datos geográficos. Algunos de ellos utilizan software que, a través de una API, permiten a los usuarios crear aplicaciones personalizadas. Estos servicios ofrecen por lo general callejeros, imágenes aéreas o de satélite, geocodificación, búsquedas en nomenclátores o funcionalidades de enrutamiento.

El desarrollo de Internet y las redes de comunicación, así como el surgimiento de estándares OGC (metadatos de información geográfica, mapas compuestos por capas, servicios web) que facilitan la interoperabilidad de los datos espaciales, ha impulsado la tecnología *web mapping*, con el surgimiento de numerosas aplicaciones que permiten la publicación de información geográfica en la web. De hecho, este tipo de servicios web mapping basado en servidores de mapas que se acceden a través del propio navegador han comenzado a adoptar las características más comunes en los SIG tradicionales, lo que ha propiciado que la línea que separa ambos tipos de software se difumine cada vez más.

Red de sensores inalámbrica

Las redes de sensores consisten en la composición de pequeños dispositivos provistos de sensores, cada uno de los cuales se denomina nodo, y todos ellos se agrupan para realizar una tarea común. Cada uno de los dispositivos mide factores de su entorno, bien sean de humedad, temperatura, luminosidad…, y envían a un procesador central los datos recogidos para ser tratados. El conjunto de sensores que realizan un mismo fin y se comunican entre ellos mediante protocolos de comunicación, sin cables, se denomina red de sensores inalámbrica.

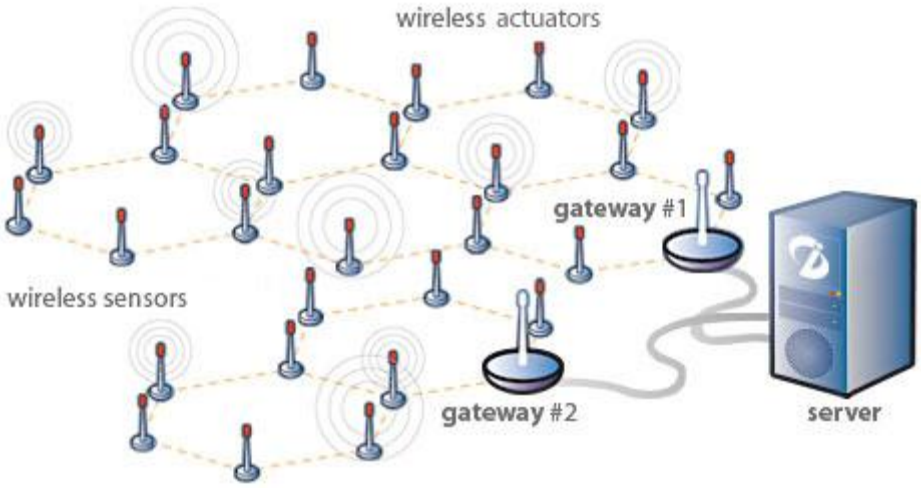


Ilustración 5: Ejemplo de red de sensores inalámbrica

Este tipo de redes están concebidas para enviar los datos medidos a una unidad central de procesamiento mediante el uso de distintas tecnologías sin cable. Las tecnologías más utilizadas en la comunicación de los sensores inalámbricos son el Bluetooth, la identificación de la frecuencia de radio (RFID) o el estándar IEE 802.11.

Los nodos de la red deben ser capaces de ejercer tanto de emisores como de receptores, del mismo modo, el nodo que actúa de puente con el servidor que procesa los datos también actúa de emisor y receptor. En la siguiente imagen podemos ver el pequeño tamaño de lo que sería un ejemplo de nodo, que ocupa una superficie menor a la palma de la mano.

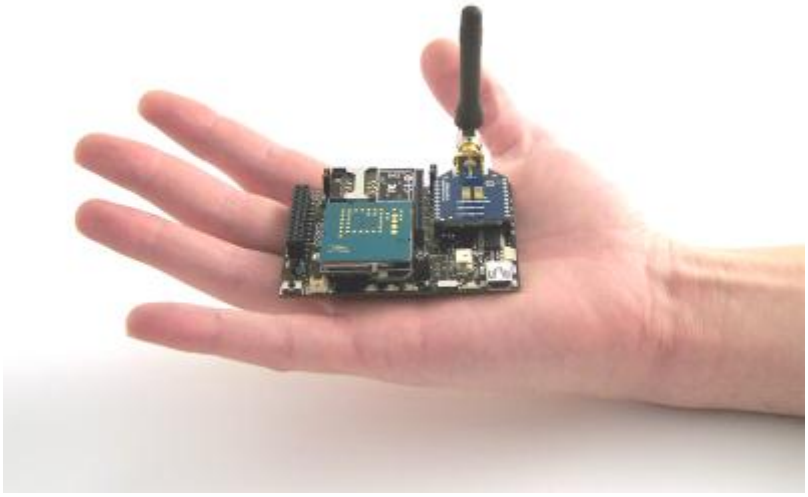


Ilustración 6: Sensor inalámbrico

Este tipo de redes de sensores se caracterizan por su sencillez a la hora de desarrollarlas e implantarlas, puesto que como hemos dicho anteriormente, cada uno de los nodos puede ser configurado para realizar tareas de emisor, receptor o de enrutador. La red inalámbrica de sensores consta de numerosas virtudes, algunas a destacar son:

- **Uso eficiente de energía.** Son redes que necesitan de un consumo mínimo de energía para su funcionamiento, lo que permite una mayor autonomía.
- **Alta disponibilidad.** Este tipo de redes están configuradas mediante algoritmos de enrutamiento dinámico, lo que permite que la red siga funcionando en caso de que alguno de los nodos fallara o fuera extraído de la red.
- **Escalabilidad.** Las redes pueden ser ampliadas en cualquier momento de una manera fácil, pues si se añaden nuevos nodos a la red, el resto los reconocen automáticamente y sólo se tendrían que actualizar las tablas de rutas.
- **Fácil implantación.** Al realizar las tareas de comunicación de forma inalámbrica se facilita la implantación de este tipo de redes, posibilitando su uso en un gran número de escenarios.
- **Bajo coste.** Este tipo de nodos se fabrican masivamente, puesto que se requiere de un gran número para la red de sensores, y sus costes no son elevados, además al tener un uso eficiente de energía el consumo y los costes de mantenimiento son mínimos.

Gracias a todas estas características se permite implantar redes de sensores inalámbricas en muy diferentes entornos que ayudarán al buen mantenimiento de los espacios, como las zonas agrícolas, sociales, civiles, sanitarias…

Arquitectura red de sensores WSN

Se identifican dos tipologías de nodos en la red WSN: Nodo Central y Nodo Sensor. El Nodo Central está compuesto por la estación base radio, encargada de recopilar toda la información de la red y volcarla al sistema central de gestión, mientras que los Nodos Sensores son los que llevan integrados los diversos sensores encargados de medir los diferentes parámetros definidos.

Nodo Sensor. Son los encargados de recabar información acerca del estado de los diferentes cultivos, por lo que estarán compuestos de un módulo radio inalámbrico, junto con los diferentes sensores específicos según la zona específica del terreno.

Nodo Central. Está compuesto por la estación base WSN, encargada de recopilar toda la información de la red, un servidor con base de datos para almacenamiento de las muestras recogidas, y una conexión inalámbrica de largo alcance (3G/GPRS, WLAN, WMAN, etc.) que permita volcar la información al sistema de gestión central.

Ilustración 7: Arquitectura global del sistema

Topología de Red y encaminamiento

La topología que mejor se ajusta al escenario del proyecto, parcelas agrarias, es la de una red mallada, encargada de distribuir la información entre los distintos puntos de la explotación agraria. Esta red estaría compuesta por un Nodo Central (*Gateway*) y varios nodos sensores, que actúan como *routers*, liberando de carga de comunicación al nodo central y mejorando la comunicación dentro de la explotación. Estos nodos sensores son además puntos de medida, es decir, recopilan información de los sensores acoplados.

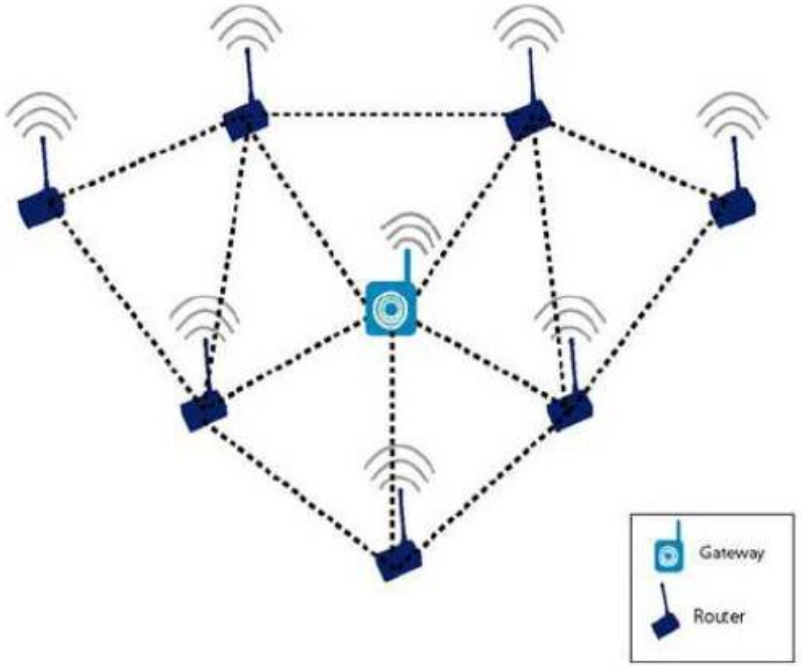


Ilustración 8: Topología en malla

Dispositivos

Según la topología establecida, se emplean los siguientes dispositivos definidos en el estándar:

- **Nodo Central**: es único en la red. Es el encargado de establecer la red inalámbrica y de coordinar a los demás nodos. Toda la información que circula en la red va a pasar a este dispositivo, el cual estará conectado a un mini PC donde se procesarán y almacenarán los datos. Este es el único elemento de la red que debe permanecer siempre activo, ya que no puede permitir la pérdida de datos.
- **Nodo Sensor:** es un dispositivo de función completa. Actúa como *router* en la red para reenviar la información y se le añade la capacidad de recogida de información, propia de los nodos finales.

Modo de funcionamiento

En el estándar IEEE 802.15.4 se indican dos modos de funcionamiento, el modo *beaconless* y el modo *beacon*. En el caso de la red planteada, el modo de transmisión empleado sería el modo *beacon*, puesto que existe la figura de un nodo central. Éste se encarga de transmitir *beacons* cada cierto tiempo para que el resto de nodos se puedan sincronizar, permitiendo en caso de necesidad una latencia mínima para aquellos dispositivos que necesiten tener este parámetro garantizado (transmisión con *beacons* y un tiempo de acceso garantizado). Como mecanismo de acceso al medio se utiliza CSMA-CA ranurado.

Metodología de formación de la red

En cuanto a la metodología de formación de red podemos ver que se mantiene una estructura en "árbol".

Un RFD (nodo sensor final) se conecta a una red en "árbol" como un dispositivo hoja al final de una rama puesto que los RFDs no permiten tener otros dispositivos asociados. Cualquiera de los nodos sensor puede actuar como un nodo central y proporcionar servicios de sincronización con otros dispositivos u otros nodos. Sólo uno de estos nodos puede ser el coordinador general, en este caso el nodo central WSN. Éste constituye el primer *cluster* eligiendo un identificador que no esté siendo utilizado y transmitiendo tramas *beacon* de forma *broadcast* a todos sus nodos vecinos. Un dispositivo que recibe una trama *beacon* podrá solicitar su unión a la red al nodo central. Si éste le permite unirse, añade al nuevo dispositivo como un hijo a su lista de vecinos. Posteriormente, el nuevo nodo añade al nodo central como nodo padre a su lista de vecinos y comienza a transmitir *beacons* de forma periódica, de forma que otro nuevo dispositivo podrá unirse a la red a través de él. Si éste último no es capaz de unirlo a la red con el nodo central, el nuevo nodo deberá buscar otro nodo padre.

1.2.2. Estudio de mercado

Las investigaciones en el campo de las redes de sensores inalámbricas predicen la llegada de nuevas generaciones de sensores inteligentes capaces de organizarse e interconectarse de forma independiente. En los últimos años se está apostando fuerte por esta tecnología y se habla de una importante revolución tecnológica de los sensores. Hoy en día existen en el mercado distintas alternativas para gestionar las redes de sensores:

- **BALMART:** es una empresa tecnológica valenciana enfocada al control inteligente del Medio Ambiente, mediante el diseño de software y la fabricación de quipos electrónicos de comunicaciones y radiofrecuencia. Su amplio abanico de ofertas incluye gestión de cultivos exteriores e interiores, control de las aguas y el medio ambiente, eficiencia en industria, ciudades inteligentes, campos de golf… Todo acerca de ellos en su página http://www.balmart.es

- **Grupo Álava Ingenieros:** este grupo de empresas con sede en Madrid, ofrece soluciones de alta tecnología en los campos de Ensayo, Medida, Comunicaciones, Seguridad, Defensa y Mantenimiento Predictivo. Disponen de instrumental diverso que abarca desde sensores para agricultura como en casos anteriores, hasta otros más específicos de desplazamiento, velocidad, posicionamiento… todo enfocado a ámbitos industriales y de seguridad. Enlace http://www.alava-ing.es/

- **GALTEL:** compañía española especializada en el diseño, desarrollo y despliegue de sistemas de comunicaciones M2M mediante la integración de redes inalámbricas de sensores y aplicaciones web para monitorización y optimización en tiempo real y análisis avanzado de los datos. Enlace http://www.galtel.es

- **SensITG Agro:** este proyecto realizado por el Instituto Tecnológico de Galicia, permite al responsable técnico del cultivo disponer de información en tiempo real y accesible vía web sobre múltiples parámetros críticos del terreno, de la vid y/o ambientales. La tecnología que utiliza permite ubicar múltiples puntos de medida con un único coste de comunicaciones.
  Enlace http://www.itg.es/?page_id=516

- **Code Blue:** desarrollado por la universidad de Harvard implementar redes de sensores en el campo de la medicina. Consiste en una plataforma hardware que permite enviar y recibir datos médicos de un paciente de forma inalámbrica.

Como conclusión al estudio de mercado, se puede afirmar que se están realizando en la actualidad, numerosas investigaciones para mejorar la tecnología de los sensores y de la comunicación entre ellos. Consecuencia de estos avances es el amplio mercado que podemos encontrar, y la competitividad entre las empresas que lo componen.

Capítulo 2 Análisis y diseño

2.1. Análisis

La aplicación a desarrollar fue pensada inicialmente para un usuario administrador, el cual cuenta con todos los permisos para interactuar con la aplicación. Estos permisos incluyen la gestión de archivos y ficheros en el servidor, la gestión de la base de datos y sus respectivas tablas, y cómo no, la gestión total de la interfaz web y sus distintas funcionalidades.

El usuario administrador puede necesitar ayuda y colaboración de personas relacionadas con el tema, por ello al administrador se le añadió una función más, la gestión de sesiones de usuario. Esto quiere decir que si el usuario administrador quiere permitir el acceso a la aplicación a estas personas podrá crear cuentas de usuario invitado. El administrador pondrá a disposición de otra persona una cuenta con usuario y contraseña, creada desde su tarea de administrador. Con esta cuenta el usuario puede acceder a la aplicación, pero no a todas las funcionalidades de las que dispone, no podrá gestionar nada del servidor ni la base de datos, ni tampoco subir ficheros desde la aplicación ni gestionar las cuentas de usuario.

Los usuario invitados verán sólo lo que quiera el administrador, si quiere que no pueda ver algún apartado, imagen… el administrador tendrá que comentar momentáneamente esa parte. Estos usuarios invitados están pensados para que trabajen en la aplicación en un corto espacio de tiempo, hasta que el administrador quiera, una vez pase este tiempo, el administrador eliminará la sesión de este invitado y el acceso a la aplicación quedará restringido.

Introducidos los actores y los distintos permisos, vamos a profundizar las funciones que pueden realizar cada uno de ellos y cómo se estructura la relación usuario-aplicación.

2.1.1. Requisitos del sistema

Existen dos tipos de requisitos:

- **Funcionales.** Describen el comportamiento que debe adoptar el sistema. Qué acciones ha de desarrollar, cómo las debe de llevar a cabo y qué resultado se debe dar para cada una de ellas.
- **No Funcionales.** Especifican cómo se debe comportar el sistema en términos de rendimiento, costo, escalabilidad… Son por tanto requisitos que no entran en la funcionalidad, pero son imprescindibles para construir una aplicación efectiva.

Requisitos Funcionales

Administrador / Invitado

- Nada más acceder a la web, aparece en un primer instante un formulario de autentificación para comprobar si el usuario y contraseña son correctos, de ser así se le permite el acceso a la intranet de la aplicación. Para ello se hace una selección de usuarios y contraseñas de la tabla de usuarios de la base de datos y se contrasta con lo pasado por el formulario de login.
- Para salir de la intranet y dejar la aplicación lo más segura posible, el usuario dispone de un apartado de logout desde el cual cerrar la sesión de usuario y volver de nuevo a la parte pública de la web que es el formulario de login.
- La interfaz web cuenta con una importante cantidad de información (mapa, menú, submenú…), por ello se establece una organización jerárquica de los mismos. De esta forma el usuario puede visualizar de manera lógica los distintos niveles en los que se divide la aplicación, así como la importancia que tienen los distintos apartados.
- El contenido de la aplicación está separado en las distintas secciones en la que se divide en menú, 8 en el caso del administrador, y 6 en el caso de usuarios invitados. Así se consigue que la información a la que se puede acceder no esté mezclada y se visualice de forma clara y concisa.
- Desde el apartado de la izquierda, el cual lista las capas que el usuario puede visualizar en el mapa, sólo hará falta seleccionarlas para que muestre aquello que el archivo del servidor contiene (parcelas, sensores, recursos hidrográficos…).
- Una vez se accede a la capa de sensores, se puede navegar entre las opciones que ofrecen de forma intuitiva y rápida, así poder ver los datos en gráficas independientes. Estas gráficas permiten visualizar los datos disponibles eligiendo las fechas de comienzo y fin, y también se pueden descargar los datos resultantes desde un logotipo de Excel en formato xls.
- El resto de objetos representados en el mapa ofrecen la posibilidad de mostrar datos relacionados con los mismos, como son descripciones, url's, datos de posicionamiento, etc.

- El usuario dispone de un botón *Home* desde el cual siempre poder volver a la página principal de la aplicación.
- Desde el apartado *Marca*, se crean archivos kml, seleccionando previamente de forma manual (en el mapa o mediante teclado) las coordenadas geográficas y posibles anotaciones. Únicamente el administrador podrá subir estos archivos al servidor.
- *Compara* ofrece la posibilidad de introducir unas fechas, y con ese intervalo temporal mostrar una gráfica con datos y porcentajes de todas las parcelas disponibles. Además se puede interactuar con la gráfica moviendo etiquetas, eliminando parcelas que no se desea comparar, seleccionando parcelas o cambiando el tipo de visualización de 2D a 3D.
- *Aemet* realiza una descarga de un archivo XML que contiene información medioambiental, seguidamente parseará este archivo para mostrarlo lo más amigable posible, con imágenes y datos organizados.
- Desde *Meteo* el usuario puede elegir entre varios campos de información y el espacio temporal deseado para así obtener información visual o digital (xls) de dicha información.
- Otra funcionalidad importante es *Qr*, desde aquí el usuario puede customizar distintos apartados, eligiendo tipo (texto, url, geolocalización…), escala, rango de errores, y de esta forma crear un logo Qr que primeramente se mostrará de forma visual aunque también ofrece la posibilidad de descargar.
- El administrador es el único que puede acceder a la información de la base de datos, realizar inserciones y modificar datos. La base de datos tiene una contraseña que solo conoce el administrador, para así tener la información controlada.
- El mapa ofrece la posibilidad de aumentar o disminuir el zoom para así ver con más detalle una zona especificada. Además también posee la opción de cambiar el tipo de vista en la que se desea ver el mapa (satélite o mapa).
- Los títulos que se encuentran en la interfaz son intuitivos y útiles para el usuario, de forma que el usuario no necesita una formación previa.
- En la parte inferior encuentran información del desarrollador de la aplicación, el cual es el encargado del mantenimiento y actualización de la misma, podrán contactar con él para solucionar posibles problemas, aunque el desarrollador tendrá que comunicar al administrador antes de realizar los cambios.

Administrador

- Puede dar de alta nuevos usuarios, que se comportarán como usuarios invitados. Lo que principalmente se realiza con esto es una inserción en la base de datos, tras lo cual se envía un mensaje por pantalla con el resultado de la inserción (éxito o fallo).
- Otro punto es dar de baja a los usuarios invitados, para ello dispone de una lista con los nombres los cuales sólo deberá clicar los que desee y Aceptar, para así realizar un borrado en la tabla de la base de datos. Este hecho también avisará del éxito o no de la operación.

- Puede subir una nueva capa que aparecerá en el Home de la página web, tanto para el administrador como para los usuarios invitados. Al realizar esta subida, lo que se está haciendo es subir por FTP el archivo al servidor web y además realizar una inserción en una tabla con las capas que serán utilizadas por la interfaz.
- Al igual que la inserción y subida de archivos KML, el administrador podrá seleccionar aquellas capas que no desee utilizar y eliminarlas de la web, las desecha eliminándolas de la tabla de la base de datos.

Requisitos no funcionales

Estos requisitos son implícitos al sistema pero no son funcionalidades concretas a este. Se descomponen en distintos apartados:

Calidad

Para el correcto funcionamiento de la web es necesario que esta se mueva en unos márgenes de calidad adecuados. Analizamos algunos de los más importantes.

- ***Integridad y seguridad:*** Será necesario asegurar que personas ajenas a la aplicación no puedan acceder a ella. Para aquellos usuarios que tengan privilegios de acceso habrá que corroborar que su acceso no supere el nivel donde los permisos tengan vigencia. Además los datos del programa deben ir cifrados para dar seguridad al protocolo http.
- ***Flexibilidad y portabilidad:*** Al ser una web con elementos genéricos, debe ser fácilmente adaptable a nuevos escenarios. Por tanto, el software debe ser ampliable y modificable. Además debe ser portable para poder aprovechar partes de la infraestructura, como por ejemplo la base de datos.
- ***Eficiencia y fiabilidad:*** Tanto el diseño como la implementación deben tener la mayor calidad posible, de forma que no se usen recursos excesivos y éstos además presenten los menos fallos posibles. Además de esto, la aplicación debe estar siempre disponible para que el usuario pueda acceder las 24 horas del día, esto dependerá externamente del funcionamiento del servidor web.
- ***Interoperabilidad:*** Será posible unir el programa a otros, sin realizar grandes cambios en el sistema.

Carga

Se establece como requisito no funcional lograr unos parámetros de carga razonables. Por una parte, se podrán conectar a la aplicación tantos usuarios como usuarios estén registrados en la base de datos. Además a dicha base de datos se accede con un nombre de usuario y contraseña para realizar la conexión, pudiendo conectarse varios usuarios de la web con la misma cuenta de la base de datos al mismo tiempo, que es lo que se pasa en este caso. De manera distinta se podría usar una cuenta de base de datos por usuario de la aplicación.

Otro aspecto a tener en cuenta será el tiempo de respuesta. Si por cada acción que realiza el usuario debe esperar un tiempo demasiado grande, la aplicación dejará de ser ágil y por tanto útil. El tiempo máximo de respuesta del servidor es de 10 segundos, tras los cuales no servirá nada a excepción de un error de tiempo de espera excedido.

Por último, se pretende evitar que el número de errores en la aplicación sea descontrolado. Para ello, ya que no se pueden controlar de forma automática, deberemos realizar pruebas de errores continuamente, tanto a la hora de realizar el programa como en un posterior análisis, así el porcentaje de errores irá siendo cada vez menor.

Coste

Se presenta el coste como un factor importante del desarrollo. Se desea construir una aplicación cuyo coste sea bajo, tanto a la hora de implementarla, como a la hora de mantenerla. En este sentido es interesante el uso de software libre para lograr este propósito. Para ahorrar coste en el desarrollo de la aplicación web, se ha optado por un servidor y un dominio gratuito, a diferencia de otros con un elevado coste mensual. Además todos los recursos utilizados son de código abierto con lo que se consigue el uso compartido de los mismos, ahorrando coste monetario y temporal.

Requerimientos tecnológicos

Se debe tener en consideración que la aplicación debe ser accesible desde cualquier sistema operativo. El sistema usará el protocolo http para conectar los clientes y el servidor. Por tato, será necesaria una máquina desde la que controlar el servidor y un navegador web en aquellos equipos donde se sitúen los clientes. El navegador podrá funcionar con cualquier resolución igual o superior a 640 x 480 píxeles.

Para lograr independizar la aplicación de plataformas o usuarios se hará uso del lenguaje UML (Lenguaje Unificado de Modelado) en el diseño y lenguajes como php, html, JavaScript, etc. en la implementación.

Interfaces

Las pantallas que podrá consultar el usuario deberán ser intuitivas y sencillas de usar. La información deberá ser presentada de forma clara y ordenada. La lógica de navegación deberá ser simple y por tanto los botones tendrán que tener comportamientos poco complejos. En resumen, se buscan interfaces amigables y simples.

2.1.2. Casos de uso

Un caso de uso es una técnica usada para realizar la captura de requisitos de un sistema informático. Cada caso de uso muestra los escenarios que indican cómo se interactúa entre el sistema y los usuarios del mismo, sin presta atención a cómo se realizará su desarrollo. En este caso tenemos dos tipos de usuarios diferenciados:

1. **Administrador del sistema**: Aquella con los permisos necesarios para llevar a cabo las funciones administrativas y con acceso a la base de datos para sus posibles modificaciones. Además de visualizar, puede gestionar archivos y dar de alta y baja usuarios invitados.

2. **Invitado**: Aquella con permisos únicamente de visualización y creación de archivos para su propio uso.

2.1.2.1. Gestión aplicación web

Los siguientes tres diagramas muestran las dos primeras iteraciones de los casos de uso de la gestión de la aplicación web. Estos diagramas serán los más generales, y se irá dando una definición más exacta del sistema a desarrollar según se vaya avanzando en las iteraciones. Para este sistema en concreto, con realizar el diseño en cuatro iteraciones es suficiente para obtener lo que será el sistema implementado.

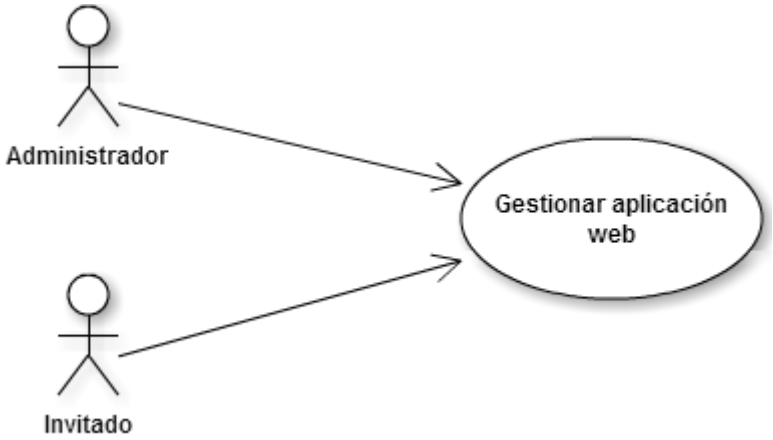


Ilustración 9: Caso de uso "Gestionar aplicación web" en Draft 0

En la imagen anterior se pueden ver los actores nombrados con anterioridad. Tanto el administrador como el usuario invitado van a ser los únicos capaces de gestionar los diferentes apartados de la aplicación web. Cada uno de los usuarios que acceda al sistema únicamente puede gestionar todo aquello derivado de su rol.

Como se muestra en la siguiente ilustración, el administrador tiene la doble funcionalidad de gestionar la web accediendo a través de su url, y de gestionarla también accediendo a la base de datos y actualizando valores. El usuario invitado sólo podrá acceder a la aplicación web e interactuar con los apartados predispuestos para él.

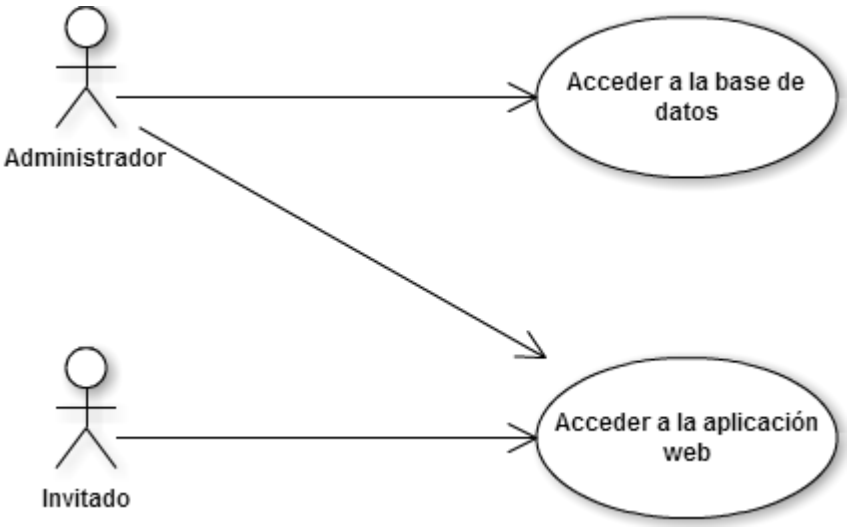


Ilustración 10: Caso de uso "Gestionar aplicación web" en Iteración 1

A continuación vamos a tratar los casos de uso que siguen al acceso a la interfaz principal. Aquellos casos referentes a la base de datos no serán tratados puesto que no atañen directamente a los usuarios de la página web.

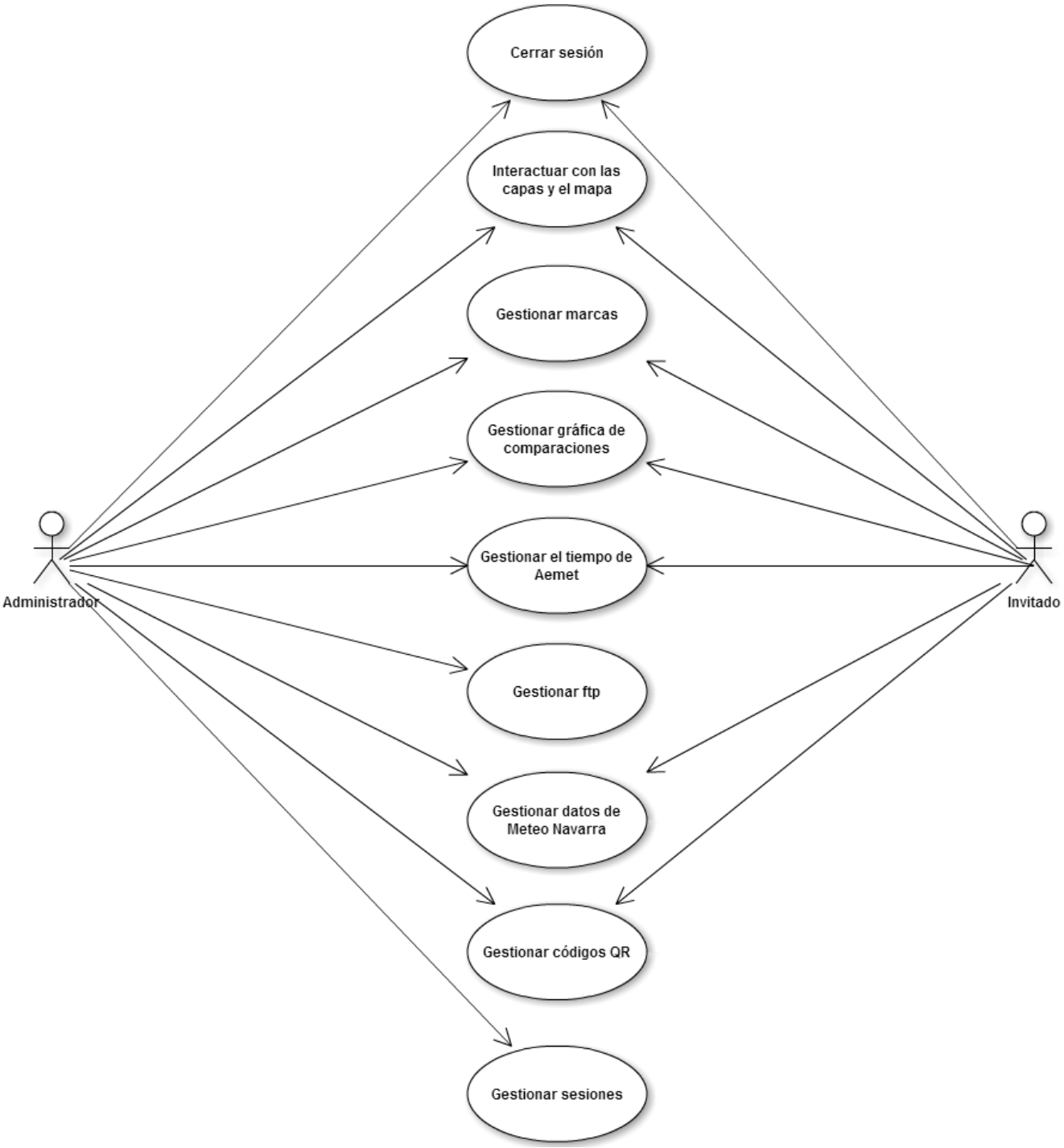


Ilustración 11: Caso de uso "Gestionar aplicación web" en Iteración 2

Podemos observar cómo una vez superado el control a la aplicación, los distintos actores encuentran distintas formas de interactuar con el sistema, dependiendo de los permisos con los que cuenten. Estos casos de uso son los que se definirán con más detalle a continuación.

- **Interactuar con las capas y el mapa**

Este caso de uso es el perteneciente al apartado *Home.* En esta sección, todos los actores partícipes tendrán delante de ellos la lista de las posibles capas a visualizar en el mapa, así como el propio mapa sobre el que navegar, hacer zoom… Una vez visualizada la capa de los sensores, podrán acceder a las opciones de las que disponen para ilustrar en gráficas los datos recogidos.

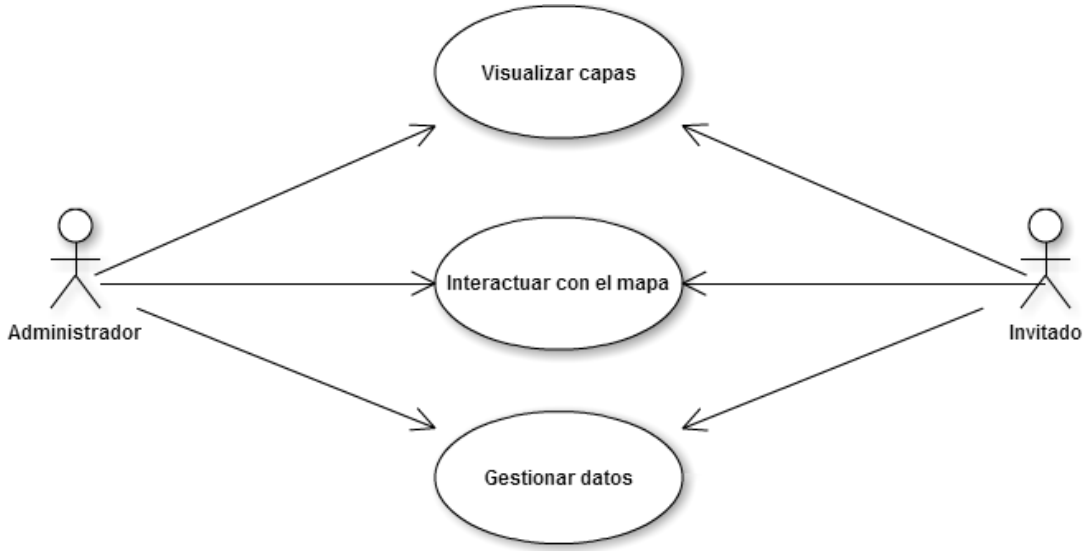


Ilustración 12: Caso de uso "Interactuar con capas y mapa" en Iteración 3

Una vez el usuario accede a los datos de los sensores, se abrirá una nueva pestaña con la que trabajar. Inicialmente aparecerán las últimas mediciones en una gráfica, y para ver los datos en un intervalo temporal disponen de un calendario y un text área en las que seleccionar las fechas. En el caso de la gráfica de barras, los usuarios en cualquier momento podrán cambiar el color de la gráfica únicamente accediendo a la paleta de colores superior. El último caso de uso que aparece en este sub-apartado es la posibilidad de descargar los datos con el botón con el icono de Excel, automáticamente los descargará al PC del usuario.

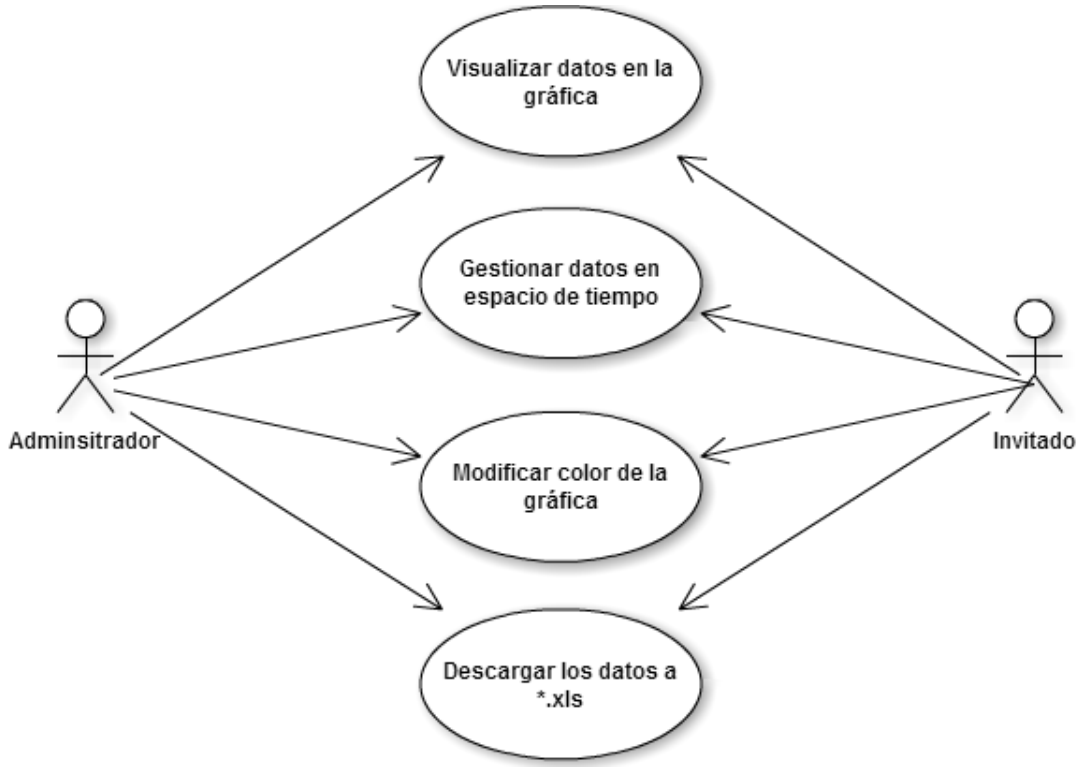


Ilustración 13: Caso de uso "Interactuar con capas y mapa" en Iteración 4

❖ Gestionar marcas

En este apartado lo que primeramente se va a encontrar el usuario es un mapa con tres opciones, una para crear marcadores, otra para crear polilíneas y la última para crear polígonos. Una vez el usuario elige la opción que se ajusta a su necesidad, creará la figura. A continuación ya podrá obtener el código KML del archivo a crear, lo tendrá que copiar y pegar en un nuevo archivo. A parte de estas opciones de crear sobre el mapa, el usuario dispone de un buscador en el que referirse a direcciones geográficas, que funciona manualmente por el usuario.

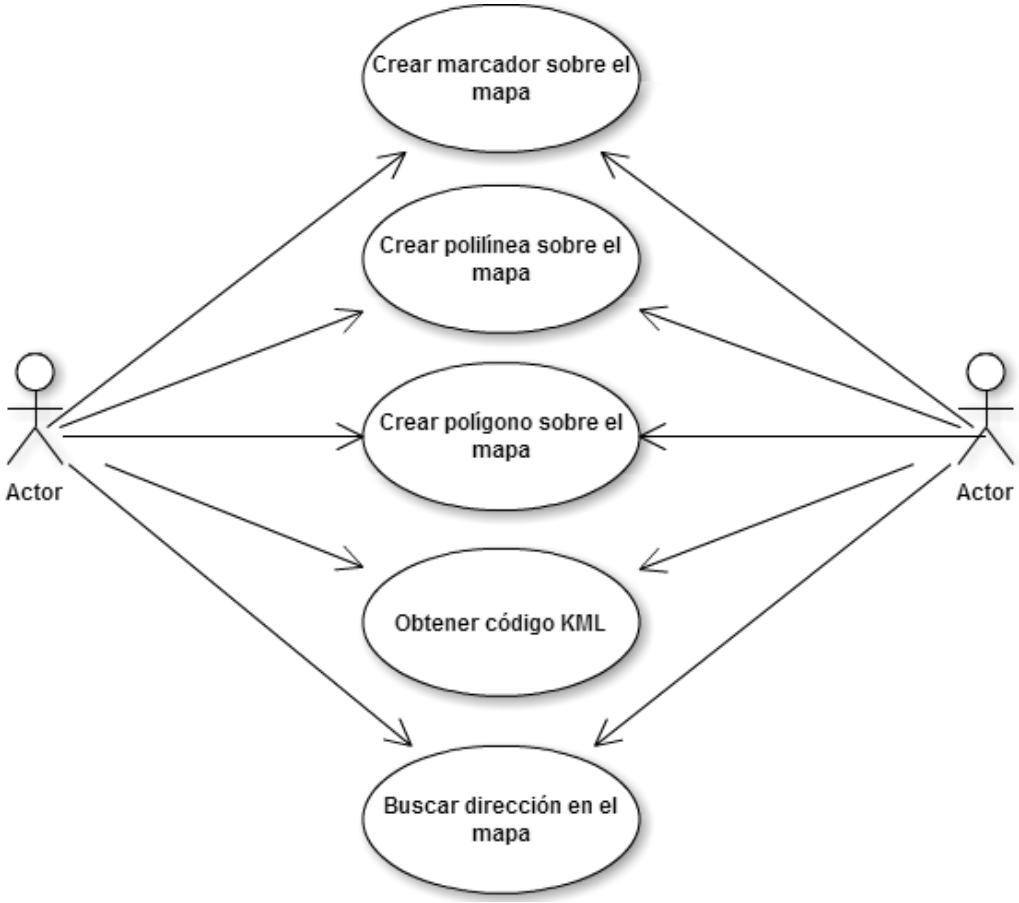


Ilustración 14: Caso de uso "Gestionar marcas" en Iteración 3

❖ **Gestionar gráfica de comparaciones**

Al igual que ocurría con los sensores que disponían de su propia gráfica individual, este apartado engloba todas las gráficas y datos de los sensores en un mismo apartado. Nada más acceder a esta nueva pestaña, cualquier usuario puede ver el resumen de los datos de todos los sensores, pues en la gráfica que se ve en primer plano estarán los datos totales con sus porcentajes sobre el total. Esta gráfica dispone de varias opciones configurables por el usuario, como la selección de sectores sobre la propia gráfica, o sobre el submenú inferior, elegir la posición de las etiquetas, el modo de visualización…

Tras interactuar con los primeros datos de la gráfica, el usuario puede cambiar las fechas entre las cuales le interesan los datos para que se le muestren en una nueva gráfica. Por último, el usuario dispone de un botón con el cual poder descargar los datos a un fichero de celdas reconocido por Excel.

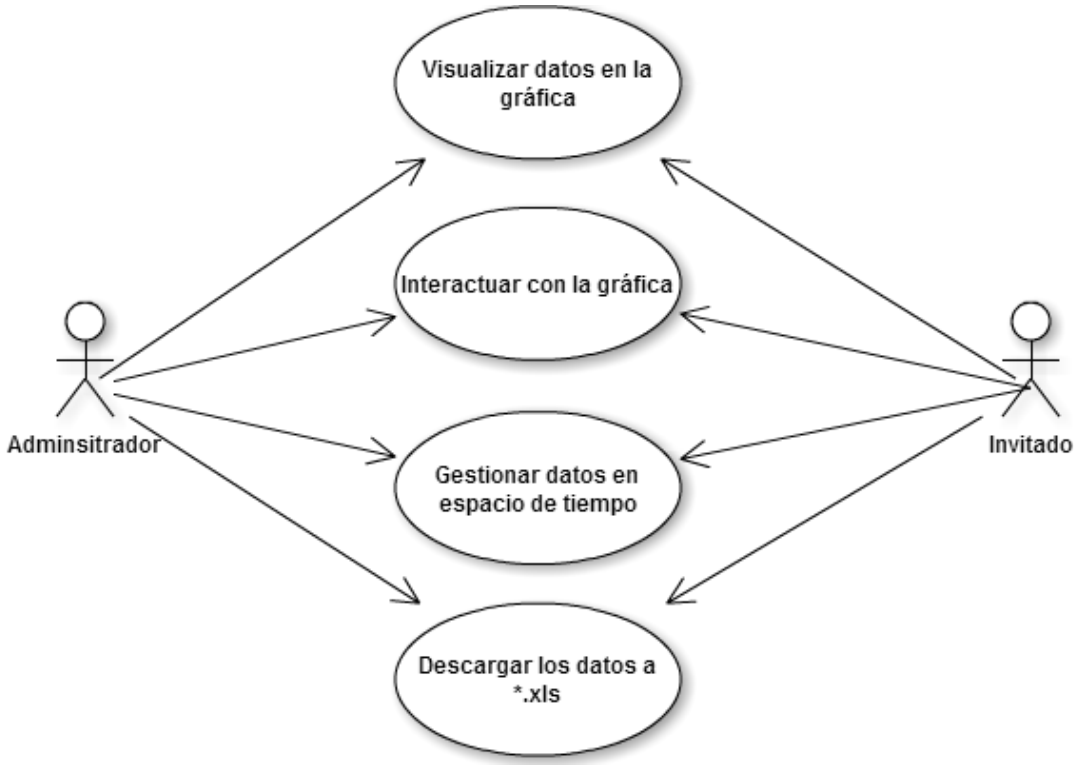


Ilustración 15: Caso de uso "Gestionar gráfica comparaciones" en Iteración 3

- **Gestionar el tiempo de Aemet**

Este apartado es totalmente visual, no tiene un fin de interactuar con el usuario, sino que únicamente actúa para mostrar información sobre futuras previsiones.

Ilustración 16: Caso de uso "Gestionar tiempo Aemet" en Iteración 3

❖ **Gestionar FTP**

El apartado de gestión de FTP sólo es accesible por el usuario Administrador, puesto que las conexiones al servidor no las puede hacer cualquiera, por motivos evidentes de seguridad. Dentro de esta gestión, los puntos principales con la subida de archivos y la eliminación de los mismos. Para realizar cambios en el servidor mediante el FTP, lo primero que se debe hacer es cumplimentar los formularios que se ofrecen y a continuación el sistema ya podrá realizar dichos cambios y actualizar el conjunto de ficheros.

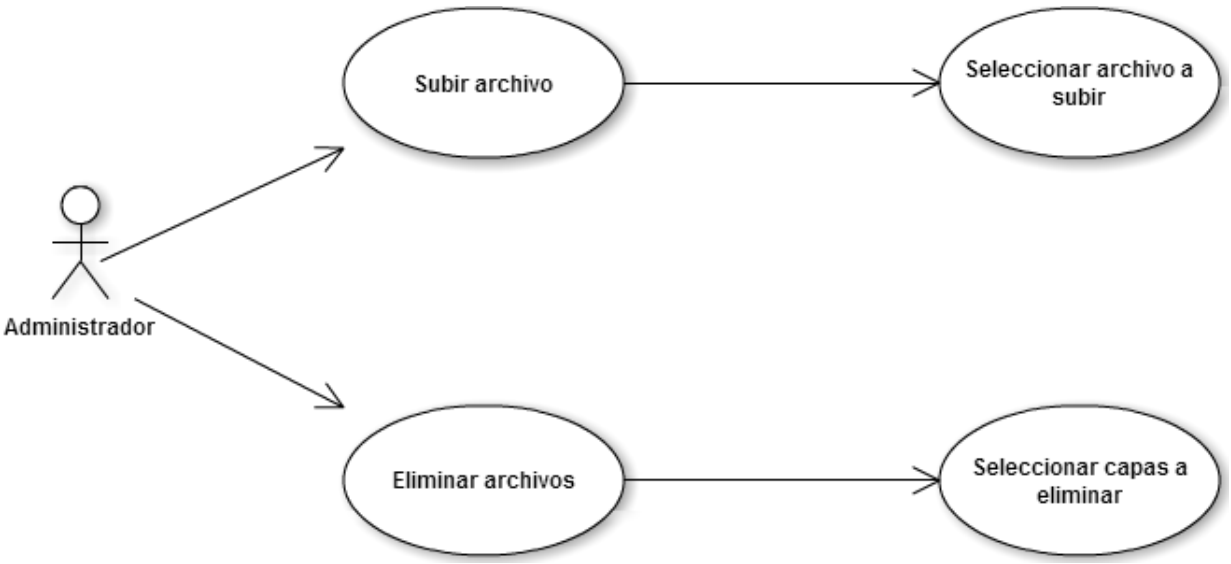


Ilustración 17: Caso de uso "Gestionar FTP" en Iteración 3

❖ **Gestionar datos de Meteo Navarra**

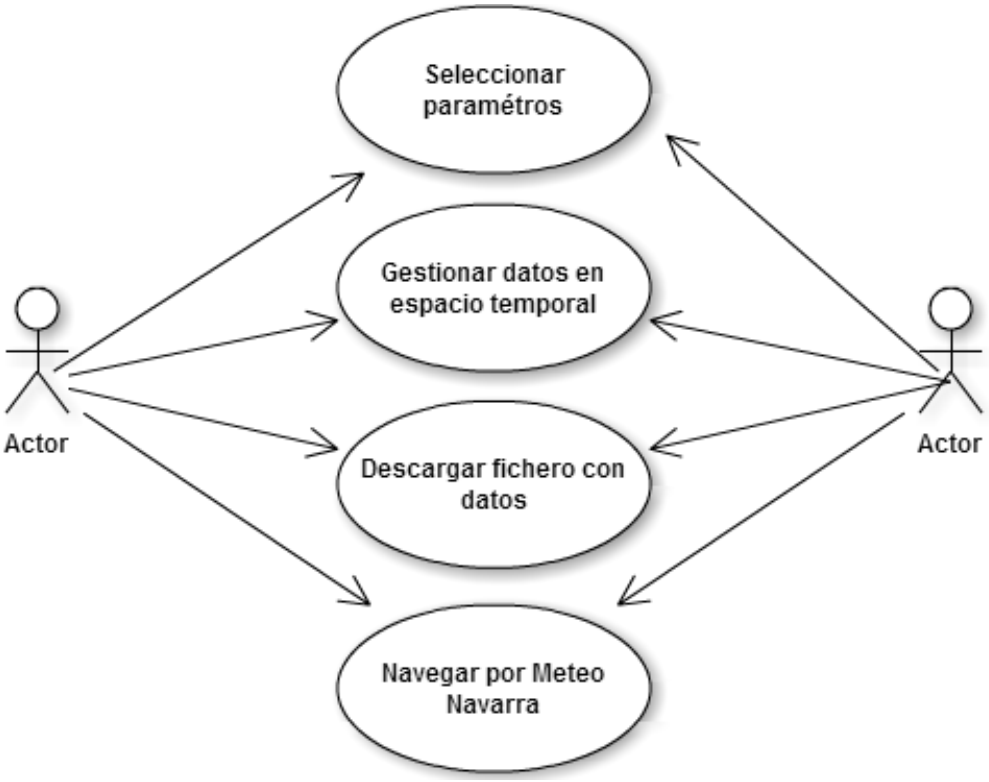


Ilustración 18: Caso de uso "Gestionar Meteo Navarra" en Iteración 3

❖ **Gestionar códigos Qr**

Este apartado vuelve a ser accesible para todos los usuarios. El usuario debe rellenar un formulario con las distintas características y datos del código Qr a realizar, y a continuación el sistema será capaz de crearlo de forma gráfica.

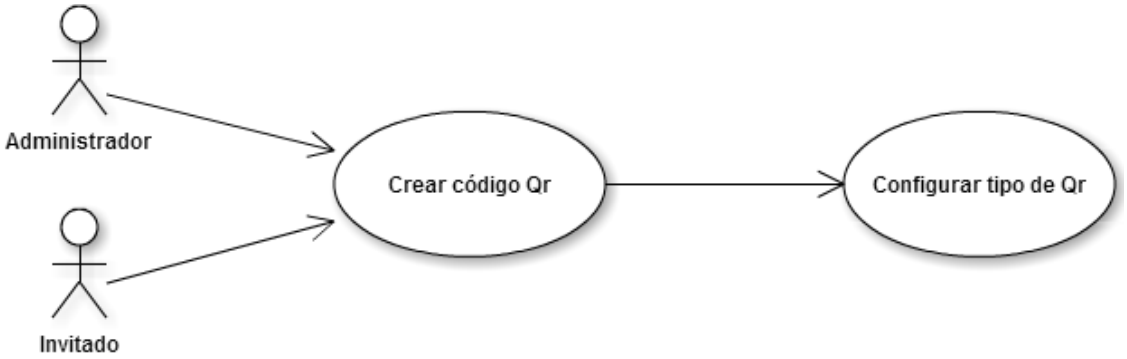


Ilustración 19: Caso de uso "Gestionar códigos Qr" en Iteración 3

- **Gestionar sesiones**

El último apartado de la interfaz también está relacionado con la gestión de la aplicación, con un único actor que será nuevamente el usuario Administrador. Las principales funcionalidades que encuentra el administrador en esta sección son dar de alta a nuevos usuario invitados, y dar de baja a usuarios. Estas acciones se llevan a cabo con la gestión de los formularios a tramitar.

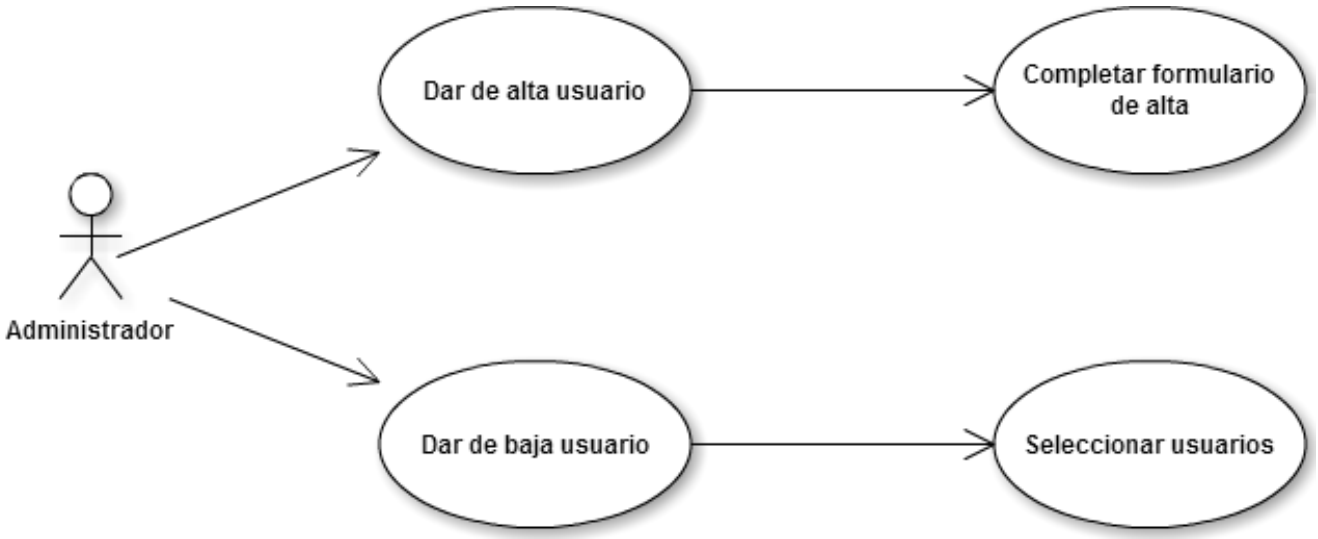


Ilustración 20: Caso de uso "Gestionar sesiones" en iteración 3

2.1.3. Modelo relacional

En este apartado vamos a aclarar cómo se gestiona la base de datos de la aplicación y por qué no es necesario realizar el modelo relacional. Para comenzar, aclarar que en este modelo todos los datos son almacenados en relaciones, esto se da en las bases de datos relacionales, en las cuales los campos se vinculan entre las tablas, mediante las claves primarias y ajenas.

Como se verá más adelante en el diseño de la base de datos, las tablas de la base de datos de la aplicación no comparten atributos. Esto se debe a que las tablas de las que se dispone son muy simples y han sido diseñadas para el mero hecho de satisfacer consultas dirigidas a una única tabla. Las tablas realizadas para la aplicación siguen la siguiente distribución

Capas (fecha, name)

Cielo (codigo, imagen)

Sensores (date, name, value1)

Users (name, password)

Viento (codigo, imagen)

Observamos que todas las tablas a excepción de la reservada para los sensores tienen dos campos, los mínimos necesarios para que uno actúa de clave primaria y el otro para aclarar algún dato como fechas, url`s, contraseñas... La complejidad podría haberse aumentado relacionando el nombre de las capas con algún nuevo campo en la tabla de usuarios, pero eso no es lo que está establecido para este proyecto. Entrando en casos de la aplicación, las parcelas y los sensores son para todos los usuarios las mismas, no se distinguen porque el administrador únicamente da permisos para poder ver lo que él ve, sin realizar modificaciones ni especificaciones.

2.1.4. Diagrama de flujo

El Diagrama de Flujo es una representación gráfica de la secuencia de pasos que se realizan para obtener un cierto resultado. Este puede ser un producto, un servicio, o bien una combinación de ambos. En el caso de la aplicación resultante en este proyecto no se da un proceso para lograr un producto, puesto que está realizado en lenguaje web PHP, el cual no está orientado a objetos.

Los diferentes objetivos de la aplicación se basan en servicios. El proceso para lograr estos servicios es bien sencillo, y únicamente interrelaciona al usuario con la página web de esta manera:

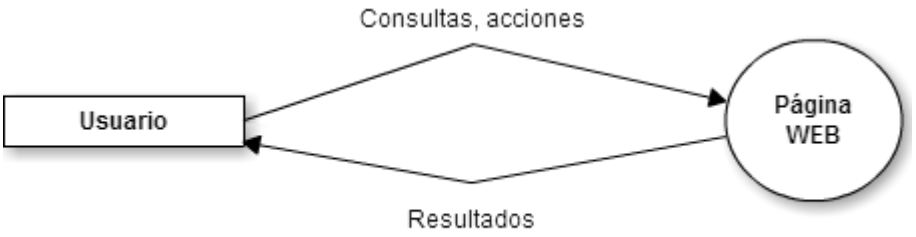


Ilustración 21: Diagrama de flujo

Los distintos flujos que se mueven entre el usuario y la página web son de acción-reacción, el usuario realiza una consulta o una acción dentro de la interfaz, y acto seguido el software da una respuesta dependiendo de la acción, pero sin seguir un esquema ramificado con distintas alternativas.

2.2. Diseño

2.2.1. Metodología de trabajo

Para diseñar este proyecto se ha seguido el modelo en **Espiral de Boehm**. Es un modelo de desarrollo de software evolutivo, las actividades a realizar no están fijadas en un principio, sino que cada cierto tiempo se consulta con el cliente. El primer paso consiste en recolectar los requisitos que el tutor te da, y realizar una planificación inicial del proyecto. A continuación, se analizan los posibles riesgos y problemas que pueden surgir a la hora de desarrollarlo, la primera vez no se analiza mucho puesto que todavía no se tiene una idea definida del resultado y se procede a realizar las primeras pruebas.

El proceso de desarrollo no debe ser de una duración muy prolongada en el tiempo, puesto que después de realizar una primera versión, el cliente (tutor en este caso) debe verla y en función de lo que él quiere la evalúa y realiza una serie de comentarios para mejorarla. Con estos comentarios comienza una nueva vuelta a la espiral, tiendo ahora como requisitos para planificar la primera versión realiza y los comentarios del cliente. Seguido se evalúan los riesgos, pero esta vez teniendo en cuenta la reacción del cliente y se procede a desarrollar una segunda versión.

Con las continuadas evaluaciones y consultas al cliente, se consigue que el prototipo esté cada vez más refinado hasta que llega el momento en el que el cliente da el visto bueno y se consigue el prototipo final. La interacción con el cliente debe ser lo más usual posible, porque como se dice “el tiempo es oro”, y si se invierte mucho tiempo en desarrollar una versión y luego el cliente te la echa para atrás porque no es lo que él quería, no habrá servido para nada el esfuerzo y trabajo.

Una vez el proyecto está finalizado, el desarrollador del mismo sólo se encargará del mantenimiento del software hasta su retirada del mercado. A continuación vemos los pasos anteriormente descritos en una ilustración, se puede observar cómo va realizando distintos ciclos y refinando el producto a desarrollar.

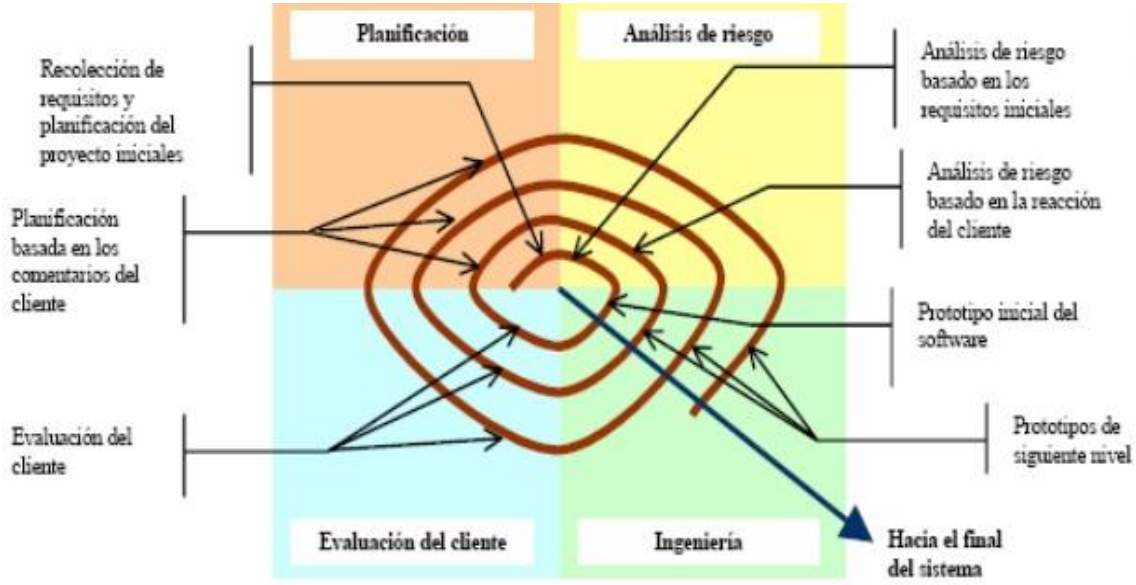


Ilustración 22: Espiral de Boehm

2.2.2. Arquitectura

A continuación vamos a ver cómo está estructurada la aplicación. A la hora de separar las partes principales encontramos tres secciones diferenciadas, una **estructura en tres capas**. Con este estilo de programación, ramificamos el proyecto en una capa de presentación, otra de negocio y la última la capa de datos.

La **Capa de Presentación** es la que ve el usuario, presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso. Esta capa se comunica únicamente con la capa de negocio. También es conocida como interfaz gráfica y debe tener la característica de ser “amigable” para el usuario.

La **Capa de Negocio** es la que recibe las peticiones del usuario y se encarga de mandar las respuestas tras el proceso. Se denomina capa de negocio porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos que almacene o recupere datos.

La última capa, la **Capa de Datos**, es donde residen los datos y es la encargada de acceder a los mismos. Está formada por un gestor de base de datos que realiza todo el almacenamiento de datos, recibe las solicitudes de almacenamiento o recuperación de información de la capa de negocio.

Las aplicaciones web como ésta suelen tener esta estructura. En la capa de presentación está el navegador que permite visualizar la página web, se comunica con el servidor web que conforma la capa de negocio, y posteriormente se accede a la base de datos. A continuación vemos en forma de diagrama como es esta estructura y las comunicaciones entre las capas que la componen.

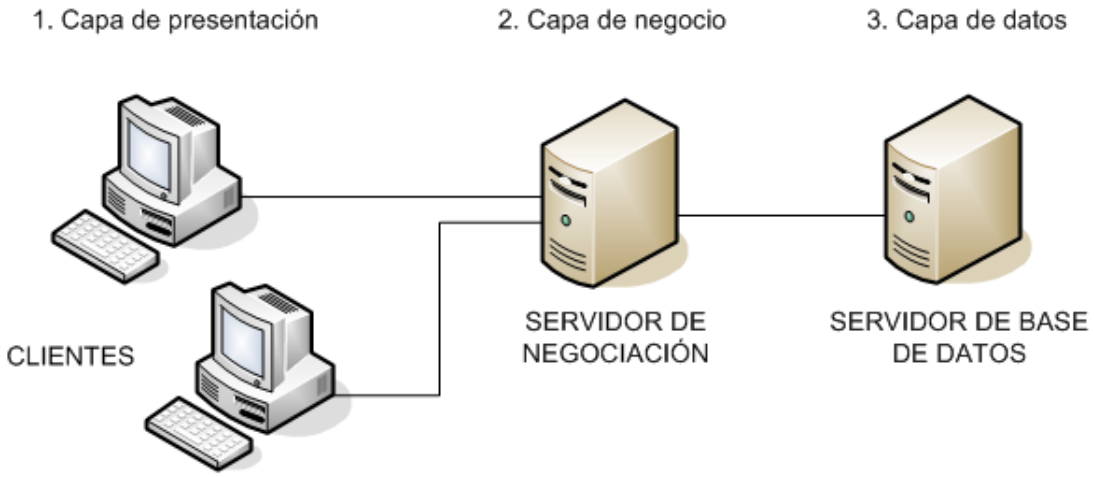


Ilustración 23: Arquitectura en 3 capas

Una vez separadas las distintas partes de la aplicación en niveles vamos a especificar un poco más la arquitectura de la aplicación web, se la conoce como **Arquitectura Clásica**. Esta arquitectura se basa en el modelo **Cliente/Servidor**.

Como todo sitio web, tenemos el navegador en la parte del cliente, el servidor web en la parte del servidor y una conexión de red para comunicar ambas partes. El cliente realiza peticiones otro programa, el servidor, quien le da respuesta. En el caso de esta aplicación, el sistema es multiusuario, teniendo el usuario centralizado en una máquina y el servidor en otra, aunque ambos actores pueden estar en la misma.

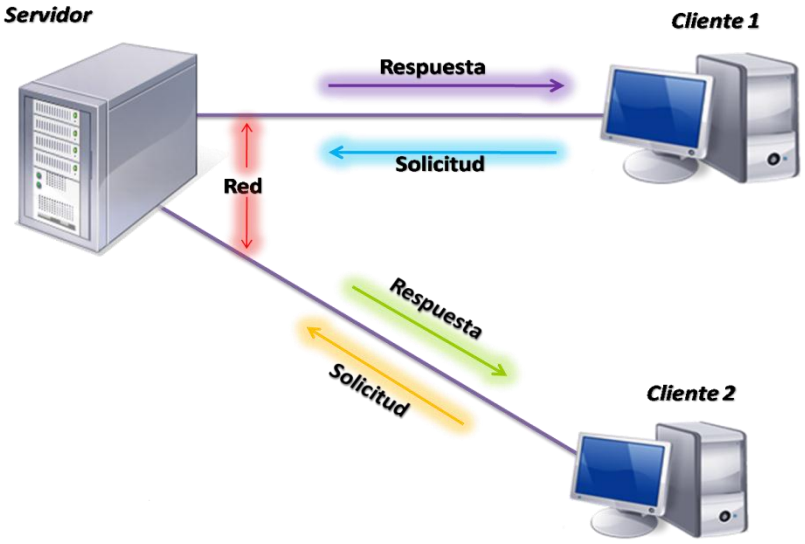


Ilustración 24: Esquema cliente-servidor

En esta arquitectura, los procesos están repartidos entre los clientes y los servidores, de tal modo que la información está centralizada en el servidor y la pone a disposición del cliente cada vez que éste la solicita. El cliente es el usuario que utiliza la aplicación, quien realiza solicitudes y peticiones al servidor, como que muestre datos pedidos por el usuario o realizar acciones sobre la propia web. La respuesta del servidor es esperada y recibida por el cliente de forma gráfica mediante la interfaz de la aplicación. El tiempo entre la petición y la respuesta depende de la velocidad de conexión de la que dispone el cliente.

El servidor actúa de forma pasiva en la comunicación con el cliente, un estado constante de espera a la llegada de solicitudes. Tras la recepción de una solicitud, la procesa y luego envía la respuesta, esto pasa por ejemplo cuando el cliente ejecuta una consulta, el servidor la procesa, realiza la consulta a la base de datos y devuelve la información al cliente, normalmente con un tiempo de respuesta muy breve. En el caso de que haya más de un usuario conectado a la aplicación, al servidor, este contesta las distintas peticiones mientras no se superen el máximo de conexiones permitidas.

2.2.3. Base de datos

Descripción de tablas

- **CAPAS**: Esta tabla contiene los nombres de todas las capas que se pueden visualizar en el mapa de google maps, así como la fecha en la que fueron introducidos en el sistema. Con esta tabla se consigue que el usuario pueda ver listadas las capas, tanto en el apartado principal Home, como en la sección de eliminación de capas. Esta tabla contiene como clave primaria el atributo 'name', puesto que no tendremos dos capas con el mismo nombre.

    - **Fecha**: Almacena la fecha en la cual se ha introducido una determinada capa KML en el servidor dela aplicación. Como todas las fechas, es un campo del tipo *Date*. No puede tomar valores nulos.

    - **Name**: Es la clave primaria de la tabla, luego no puede tomar valor nulo. Contiene el nombre identificador del archivo KML. Al ser un identificador, podrá contener caracteres alfanuméricos, por tanto será del tipo *Varchar* con una amplia longitud de treinta caracteres.

- **CIELO**: Esta tabla es de gran uso para realizar el parseo del archivo *.xml de meteorología. En ella se encuentran los códigos identificadores de los diferentes estados posibles del cielo, y se asocian con una dirección url que representa ese estado. Su clave primaria es el identificador del código.

    - **Código**: Es la clave primaria de la tabla, no admite valor nulo. Este atributo contiene caracteres numéricos que pueden ir seguido de una letra. Según el código que proporciona el archivo *.xml, se refiere a un estado del cielo u a otro. Por ejemplo el código 11 significa "cielo despejado" y el 15n "cielo muy nuboso". Es del tipo *Varchar* con longitud de cuatro.

    - **Imagen**: Contiene la dirección url del servidor donde está la imagen que representa el estado del cielo que pertenezca al código. De esta manera cuando se encuentre un código en el fichero, con una simple consulta a esta tabla obtendremos su significado. No puede tener valor nulo porque todo código necesita tener una imagen asociada, pertenece al tipo *Varchar* con longitud 100, para que pueda entrar una dirección url larga.

- **SENSORES**: Esta la tabla más importante y de más uso de la aplicación, puesto que es la tabla donde los sensores introducen los valores recogidos en las mediciones. Es la tabla más poblada, puesto que contiene la fecha, el nombre del sensor y el valor de la medición de cada sensor. Posee dos claves primarias, la fecha y el nombre del sensor, puesto que en una fecha puede haber mediciones de distintos sensores y cada sensor tendrá medidas en fechas distintas, además que los valores pueden ser cualesquiera.

    - **Date**: Es el campo encargado de recoger la fecha en la que se realiza una medición, tipo *Date* como todas las fechas. No puede tomar valor nulo porque es clave primaria.

    - **Name**: Recoge el nombre del sensor junto al nombre de la parcela donde se encuentra, para una mayor aclaración. Junto a la fecha es la clave primaria, el conjunto de ambos es único (fecha-nombre), por ello no puede tomar valor nulo. Pertenece al tipo *Varchar* de longitud treinta y dos.

    - **Value1**: Contiene el valor de cada medición de los sensores, un valor mayor a 0. El valor debe existir, luego no admite valor nulo. Para abarcar un gran rango de números, se ha decidido que sea un *Varchar* de longitud quince.

- **USERS**: Esta también es una tabla de gran importancia, por su delicado contenido. Contiene los nombres de usuario y sus respectivas contraseñas. A esta tabla se accede para comprobar la autenticidad del usuario y permitir el acceso a la aplicación, y también en el apartado de sesiones (modo administrador) para dar de alta o baja a un usuario invitado. Como clave primaria utiliza el identificador de usuario, que siempre será único.

    - **Name:** Atributo identificativo de cada usuario. Tendrá una longitud máxima de diez caracteres, de tipo *Varchar*. Es la clave primaria de la tabla, puesto que no habrá dos usuarios con el mismo nombre, no se acepta el valor nulo.

    - **Password:** En este campo se guardará la contraseña de los distintos usuarios. La contraseña se guarda encriptada en la base de datos, para que si alguien con fines malignos accede a la tabla no pueda conseguir las claves. Por estos motivos una contraseña relativamente corta, en md5 llegará a ocupar treinta y dos dígitos hexadecimales (*Varchar*). Es obvio que este campo tiene que tomar algún valor.

- **VIENTO:** Junto a la tabla *CIELO*, esta tabla se encarga de la realización del widget del tiempo de la aplicación web. A la hora de parsear el archivo *.xml, se obtienen las direcciones de los vientos de los distintos días, y tras esto se hace una consulta a esta tabla, buscando la url de la imagen que corresponde a cada dirección. El código de la dirección actuará como clave primaria, ya que los códigos serán las distintas direcciones geográficas (N, S, E, O…).

    - **Código:** Como se ha mencionado al describir la tabla, este atributo actúa de clave primaria, valor distinto de NULL. Únicamente contiene valores alfabéticos, con un máximo de dos caracteres, *Varchar(2)*. El código viene a ser las siglas de la determinación geográfica.

    - **Imagen:** Contiene la dirección url del servidor donde está la imagen que representa la dirección del viento asociada al código. No puede tener valor nulo porque todo código necesita tener una imagen asociada, pertenece al tipo *Varchar* con longitud 100, para que pueda entrar una dirección url larga.

2.2.4. Interfaz web

Diagrama de navegación

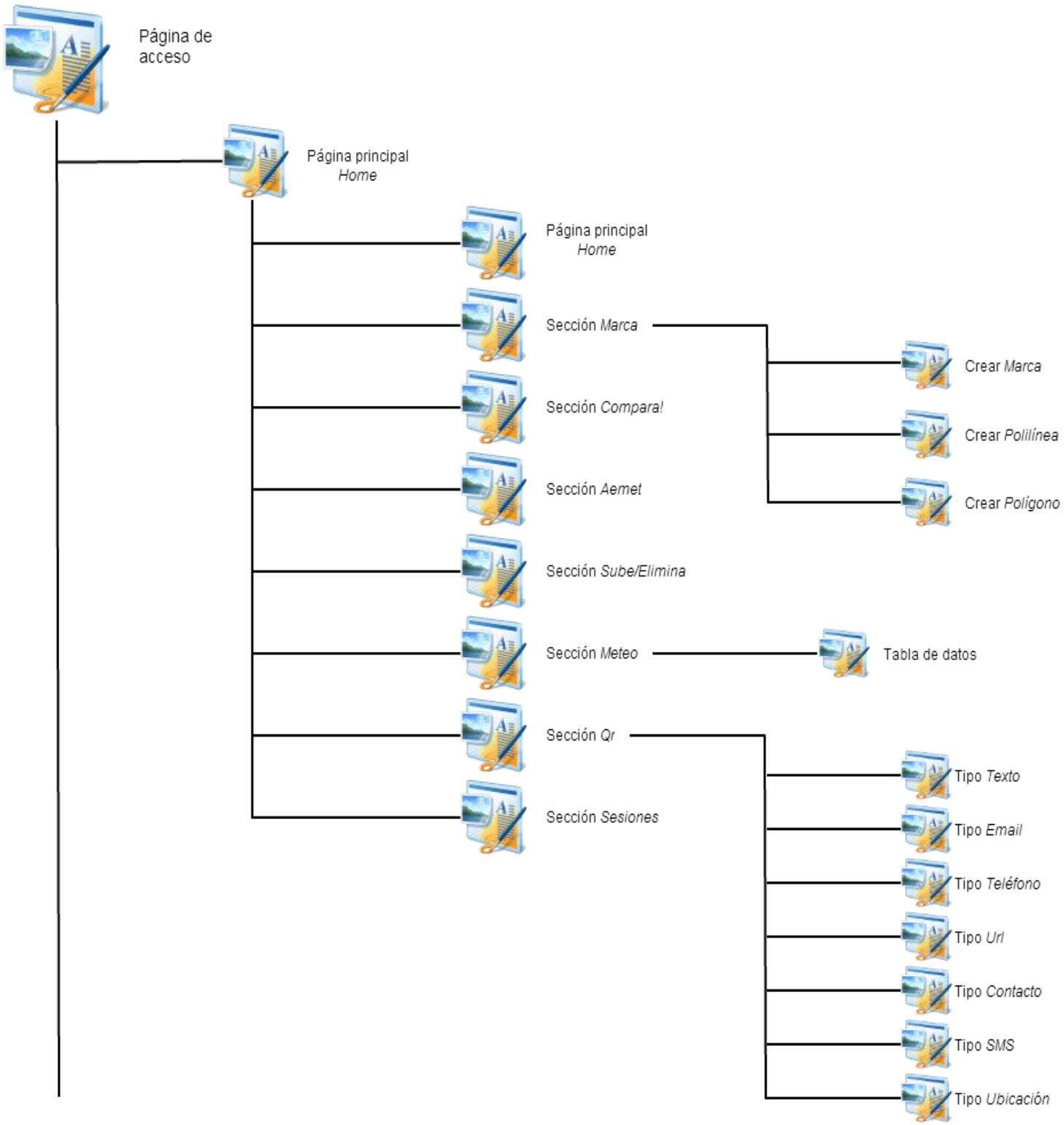


Ilustración 25: Diagrama de navegación

Diseñar el diagrama de navegación consiste en describir dinámicamente el orden y tipo de pantallas que se van produciendo durante la ejecución de la aplicación. Consiste en un autómata finito que describe la transición de ventanas para conocer de este modo las ventanas que preceden y a las que se pueden acceder desde cada una de las mismas.

En la imagen anterior se puede observar el diagrama de navegación de la aplicación realizada, de este modo se puede conocer con facilidad el número de ventanas de las que consta nuestra aplicación así como los pasos necesarios para conseguir el fin deseado. Primeramente, el usuario accede a la url de la página, donde sólo encontrará el formulario de login para acceder a la aplicación. Una vez superada la autentificación, automáticamente aparece en el navegador la página principal, o *Home*, desde la cual tendrá multitud de apartados en el menú superior para acceder. De estos apartados del menú, únicamente tres de ellos dispondrán de ventanas complementarias a la sección elegida. El primero de ellos es la sección *Marca*, desde ésta se podrán acceder a tres secciones diferenciadas y marcadas en la interfaz, una para crear marcadores, otra para crear polilíneas y la última para crear polígonos. Seguido a este apartado, el siguiente que se ramifica es el apartado *Meteo*, que conducirá al usuario lector a un nuevo iframe dentro de la aplicación, desde el cual poder ver tablas con datos reales de estaciones meteorológicas. El último apartado que nos permite una mayor navegación es la sección *Qr*, en la cual el usuario dispondrá de un menú de tipos de códigos Qr que se pueden crear, cada uno de ellos con su propia interfaz.

Capítulo 3 Implementación

3.1. Tecnologías aplicadas

3.1.1. HTML

HTML son las siglas de *HyperText Markup Language* (Lenguaje de Marcas de Hipertexto), es el utilizado para la construcción de páginas web. Su uso está destinado a describir la estructura y contenido de los sitios web en forma de texto, así como para complementar el texto con objetos tales como imágenes.

Este lenguaje se construye mediante etiquetas que delimitan las secciones de la web y alojan la información. Normalmente suele estar implementado junto con las hojas de estilo llamadas CSS que ayudan a definir los aspectos visuales de la web. HTML puede incluir scripts (JavaScript, PHP) que afectan al comportamiento de navegadores web y otros procesadores de HTML.

Puede ser creado y editado por cualquier editor de textos básico, como Gedit en Linux o Notepad++ en Windows, simplemente hay que tener en cuenta que la extensión del documento ha de ser *.html o *.htm. Dentro de estos archivos se implementará el código HTML el cual consta de varias etiquetas básicas:

- **<head>:** Define la cabecera del documento HTML, esta cabecera suele contener información sobre el documento que no se muestra directamente al usuario, como puede ser el título de la ventana del navegador.
- **<title>:** Define el título de la página que aparecerá encima de la ventana del sitio web.
- **<html>:** Define el inicio de un documento HTML, de esta forma se le indica al navegador que el texto que le sigue debe ser interpretado como código HTML.
- **<script>:** Introduce un script en una web, o se llama a uno mediante el atributo "src= *'url del script'* ".
- **<link>:** Para vincular el sitio a hojas de estilo o iconos.
- **<style>:** Coloca el sitio interno de la página, ya sea usando CSS u otros lenguajes similares.
- **<a>:** Para crear un enlace es necesario utilizar esta etiqueta de ancla junto al atributo *href*, que establecerá la dirección URL a la que apunta el enlace.
- **<body>:** Define el cuerpo o contenido principal del documento. Esta es la parte del documento que se muestra en el navegador. Dentro del cuerpo se pueden definir numerosas etiquetas, las más utilizadas son:

- **<p>:** Define un párrafo de texto en la web.
- **
:** Introduce un salto de línea.
- **<h1>:** Indica que se va a escribir una cabecera para una sección, el número indica el tamaño de la fuente, siendo el 1 el mayor.
- **<td>:** Define tablas en la página.
- **<li>, <lu>:** Definen listas de elementos, ya sean ordenadas o no.

3.1.2. Hoja de estilos CSS

CSS o lo que es lo mismo, hojas de estilo en cascada, es un lenguaje utilizado para definir la presentación de un documento escrito en HTML o XML. El W·C es el encargado de formular las especificaciones de las hojas de estilo que servirán de estándar para los navegadores.

La idea que se mantiene para el desarrollo de las hojas de estilo es separar la estructura del documento de su presentación, lo que hace más fácil la codificación de los estilos en caso de ser necesario.

Cuando se utiliza CSS, las etiquetas no deben proporcionar información acerca de la visualización del documento, sino marcar la estructura. La información del estilo, separada en una hoja de estilo, especifica cómo se ha de mostrar una etiqueta; color, fuente, alineación del texto o tamaño.

La introducción de CSS ha permitido en muchos casos reemplazar el uso de tablas, que era la técnica que se utilizaba antes para conseguir el mismo fin. Sin embargo, CSS todavía no permite la misma versatilidad pues las diferencias entre los distintos navegadores dificultan la tarea. Se espera que futuros desarrollos en CSS3 resuelvan esta deficiencia y hagan de CSS un lenguaje más apto para describir la estructura espacial de una página. Como ventajas cabe destacar:

- Los navegadores permiten a los usuarios especificar la hoja de estilo local que será aplicada en un sitio web, con lo que se obtiene una mayor accesibilidad.
- Control centralizado de la presentación de un sitio web completo con lo que se agiliza la actuación del mismo de forma considerable.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

3.1.3. PHP

El lenguaje PHP es un lenguaje de programación diseñado y orientado a la creación de páginas dinámicas. Es utilizado principalmente en interpretación del lado del servidor pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otro tipo de aplicaciones en las que se encuentran aquellas con interfaz gráfica.

La mayor fuerza de PHP reside en que es un lenguaje preparado para soportar accesos a muchos tipos de bases de datos como Oracle, ODBC, DB2, SQLServer… y enfocando nuestro caso, MySQL. Lo que hace diferente a PHP es que el código que se deba ejecutar se ejecuta siempre en el servidor, de este modo el cliente sólo recibe los resultados de la ejecución y le es imposible acceder al código que generó la página.

Las conexiones de PHP a bases de datos son enlaces SQL que no se cierran cuando termina la ejecución del script. El comportamiento de estas conexiones consiste en que al invocar una conexión, PHP comprueba su ya existe una conexión de este mismo tipo o es una conexión nueva. En el caso de que exista se procede a su uso y en caso que no exista la conexión se crea. Dos conexiones se consideran iguales cuando están realizadas al mismo servidor y con un mismo usuario y contraseña.

El código PHP se encuentra inmerso entre el código HTML y JavaScript distinguiendo entre etiquetas:

```
<?php
        código…
        ……
?>
```

3.1.4. JavaScript

JavaScript es un lenguaje basado en objetos y guiado por eventos, utilizado para acceder a objetos en aplicaciones. Se utiliza integrado en un navegador web permitiendo el desarrollo de interfaces de usuario mejoradas y páginas web dinámicas. Es un lenguaje influenciado de múltiples lenguajes y fue diseñado con el objetivo de tener una sintaxis similar a la de Java, aunque más sencilla para facilitar el uso a programadores principiantes. Todos los navegadores actuales interpretan el código de JavaScript integrado dentro de las páginas web.

Entre las acciones típicas que se pueden realizar en JavaScript tenemos dos vertientes. Por un lado los efectos especiales sobre páginas web, para crear contenidos dinámicos y elementos de la página que tengan movimiento, cambien color o cualquier otro dinamismo. Por el otro, JavaScript nos permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que podemos crear páginas interactivas con programas como calculadoras, agendas o tablas de cálculo.

Este lenguaje se puede incluir en cualquier documento y es compatible con cualquier sistema operativo. La mejor manera es incluir JavaScript como un archivo externo, tanto por cuestiones de accesibilidad como por la velocidad en la navegación, este archivo será del tipo *.js y para ser utilizado se debe escribir en el documento HTML:

```
<script type="text/javascript" src="[url del fichero js]"></script>
```

También es posible incrustar el código en el documento con la etiqueta <script>:

```
<script type="text/javascript"><!--// código JavaScript--></script>
```

jQuery

Es una biblioteca de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones (FLV) y agregar interacción con la técnica AJAX a páginas web. Fue presentada el 14 de enero de 2006 en el BarCamp NYC. jQuery es la biblioteca de JavaScript más utilizada.

jQuery es software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privativos. jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

3.1.5. AJAX

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

AJAX es una tecnología asíncrona, en el sentido de que los datos adicionales se requieren al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página. JavaScript es el lenguaje interpretado (scripting language) en el que normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante XMLHttpRequest, objeto disponible en los navegadores actuales. En cualquier caso, no es necesario que el contenido asíncrono esté formateado en XML.

AJAX es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores, dado que está basado en estándares abiertos como JavaScript y Document Object Model (DOM).

AJAX es una combinación de cuatro tecnologías ya existentes: XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información. Document Object Model (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada. El objeto XMLHttpRequest para intercambiar datos de forma asíncrona con el servidor web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto iframe en lugar del XMLHttpRequest para realizar dichos intercambios. Como el DTHML, LAMP o SPA, AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente.

3.1.6. El lenguaje KML

Definición

KML es un formato de archivo que se utiliza para mostrar datos geográficos en un navegador terrestre, como Google Earth, Google Maps y Google Maps para móviles. KML utiliza una estructura basada en etiquetas con atributos y elementos anidados y está basado en el estándar XML. Todas las etiquetas distinguen entre mayúsculas y minúsculas y deben aparecer exactamente como aparecen en la Referencia de KML. En esta referencia se indica qué etiquetas son opcionales. Dentro de un elemento determinado, las etiquetas deben aparecer en el mismo orden en el que aparecen en la referencia.

KML, o Keyhole Markup Language, es una gramática de XML y un formato de archivo para la creación de modelos y el almacenamiento de funciones geográficas como puntos, líneas, imágenes, polígonos y modelos que se mostrarán en las diferentes aplicaciones de Google (Google Mpas, Google Earth) y otras diseñadas para el mismo fin.

Cada archivo KML consta de un conjunto de elementos gráficos, imágenes y opciones de configuración. KML se utiliza para hacer lo siguiente:

- Simbolizar y mostrar datos SIG como elementos en Google Earth y Google Maps a través de símbolos, colores, imágenes y ventanas emergentes de información estilo globo.
- Permitir el acceso a la información de atributos sobre entidades geográficas, por ejemplo, con la presentación de información de atributos si hace clic en un marcador de entidad.
- Definir la interacción del usuario con aquellas entidades, por ejemplo, para controlar la configuración de la ubicación de la cámara y del destino de vuelo en Google Earth.

Google Earth procesa los archivos KML de una manera similar a como los navegadores web procesan los archivos HTM y XML. Al igual que los archivos HTMLL, los KML cuentan con una estructura basada en etiquetas con nombres y atributos utilizados para poder visualizarlos. Google Earth actúa como un navegador de archivos KML.

El modelo más simple de documento KML es el que se puede crear directamente en Google Earth; es decir, no es necesario editar ni crear ningún archivo KML en un editor de texto. Las marcas de posición, las superposiciones de suelo, las rutas y los polígonos se pueden crear directamente en Google Earth.

Estructura

El modelo más simple de documento KML es el que se puede crear directamente en Google Earth; es decir, no es necesario editar ni crear ningún archivo KML en un editor de texto. Las marcas de posición, las superposiciones de suelo, las rutas y los polígonos se pueden crear directamente en Google Earth.

Marcas de posición

Las marcas de posición (Placemark) son uno de los recursos más utilizados en Google Earth. Permiten marcar una posición en la superficie de la Tierra con un icono de chincheta amarilla. La marca de posición (Placemark) más sencilla incluye solo un elemento de punto (<Point>), que especifica la ubicación de la marca de posición. Puedes especificar un nombre y un icono personalizado para una marca de posición y, si lo deseas, le puedes añadir otros elementos geométricos. A continuación un ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
        <Placemark>
                <name><![CDATA[Parcela1_sensor1]]></name>
                <description><![CDATA[
                        <a href="http://www.parcelas.x10.mx/grafica_parcela1sensor1.ph
                        <a href="http://www.parcelas.x10.mx/grafica_sectores.php?par=p
                ]]></description>
                <styleUrl> #My_Style</styleUrl>
                <Point>
                        <extrude>1</extrude>
                        <altitudeMode>relativeToGround</altitudeMode>
                        <coordinates>-1.657113 ,42.479504,0</coordinates>
                </Point>
        </Placemark>
</kml>
```

Ilustración 26: Ejemplo de marcador KML

Este archivo tiene la siguiente estructura:

- Un encabezado XML. Es la línea número 1 de todos los archivos KML. Antes de esta línea no puede haber caracteres ni espacios.

- Una declaración de espacio de nombre de KML. Es la línea número 2 de todos los archivos KML 2.2.

- Un objeto de marca de posición (Placemark) que contiene los siguientes elementos:
    - un nombre (*name*) que se utiliza como etiqueta para la marca de posición,
    - una descripción (*description*) que aparece en una "viñeta" junto a la marca de posición,
    - un punto (*Point*) que especifica la posición de la marca de posición en la superficie de la Tierra (*la longitud, la latitud* y, opcionalmente, *la altitud*).

- Un estilo propio para el marcador, definido en otra parte de código, para así poder especificar el color, la imagen a usar…

Una marca de posición (Placemark) con un elemento de punto (Point) es la única forma de dibujar un icono y una etiqueta en el visor 3D de Google Earth. De forma predeterminada, el icono es la famosa chincheta amarilla. En KML, una marca de posición (<Placemark>) puede incluir uno o varios elementos geométricos como, por ejemplo, una cadena de líneas (LineString), un polígono (Polygon) o un modelo (Model). Pero únicamente una marca de posición (<Placemark>) con un punto (Point) puede tener un icono y una etiqueta. El punto (Point) se usa para colocar el icono, pero no hay representación gráfica del punto en sí.

Polígonos

Los polígonos (Polygon) se pueden utilizar para crear edificios simples y otras formas. Permite dibujar la estructura interna y externa, haciéndola sobresalir desde el suelo.

```
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
        <name>Finca5.kml</name>
        <Placemark>
                <name>Finca5</name>
                <styleUrl>#m_ylw-pushpin</styleUrl>
                <Polygon>
                        <tessellate>1</tessellate>
                        <outerBoundaryIs>
                                <LinearRing>
                                        <coordinates>
                                        -1.660951904889866,42.48382707932602,0 -1.660747383637582,42.48318626016147,0
                                        -1.660100476211824,42.48301596081232,0 -1.65845275395234,42.4837525243093,0
                                        -1.65902711632764,42.48467201400792,0 -1.660951904889866,42.48382707932602,0
                                        </coordinates>
                                </LinearRing>
                        </outerBoundaryIs>
                </Polygon>
        </Placemark>
</Document>
</kml>
```

Ilustración 27: Ejemplo de polígono KML

Para crear un polígono primeramente se define el elemento Polígono, <Polygon>. Seguido, mediante la etiqueta <altitudeMode>, se puede establecer la altitud del elemento con respecto a la elevación real del suelo en una ubicación concreta. Se utiliza un segmento lineal, <LinearRing>, como borde interno del polígono.

A continuación tenemos los distintos tipos de alturas a las que podemos colocar un polígono o superficie

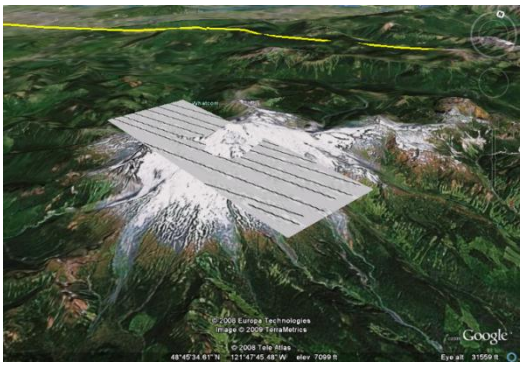


Ilustración 28: Altitud *Absolute*

1. Absolute: Mide la altitud relativa al nivel del mar sin tener en cuenta la elevación real del terreno bajo el recurso.

Ilustración 29: Altitud *clampToGround*

2. clampToGround: Ubica el recurso KML en la superficie del suelo, siguiendo el terreno.

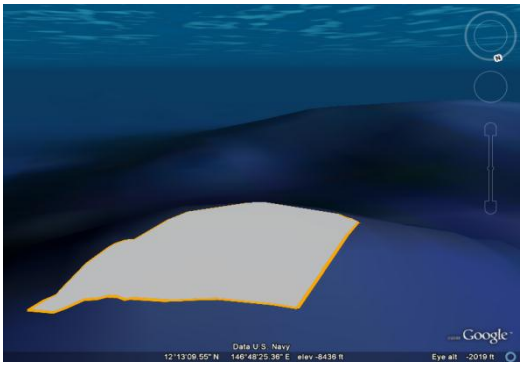


Ilustración 30: Altitud *clampToSeaFloor*

3. clampToSeaFloor: Coloca el recurso KML en el fondo de una gran masa de agua.

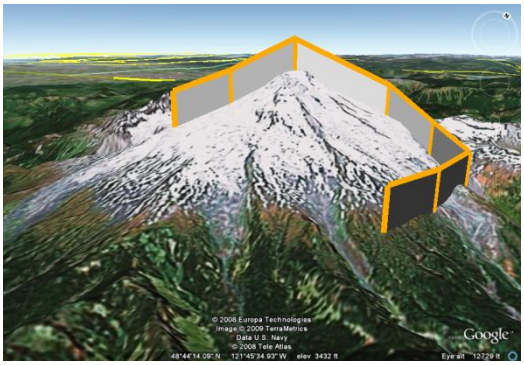


Ilustración 31: Altitud *relativeToGround*

4. relativeToGround: Mide la altitud del suelo directamente por debajo de las coordenadas.

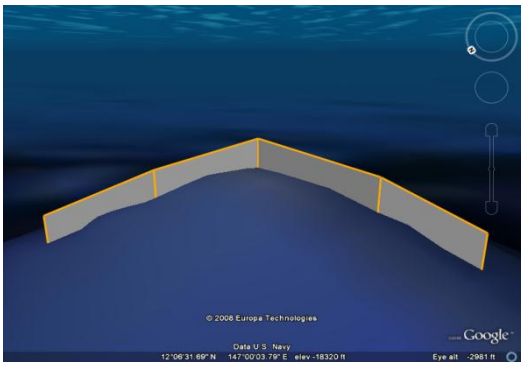


Ilustración 32: Altitud *relativeToSeaFloor*

5.- *relativeToSeaFloor*: Mide la altitud desde el nivel del fondo marino directamente por debajo del recurso.

3.1.7. Descripción del Sistema Gestor de Bases de Datos

MySQL es un Sistema Gestor de Bases de Datos relacionales, multiusuario, multihilo y desarrollado en ANSI CC y C++. Por un lado se ofrece bajo la GNU/GPL para cualquier uso compatible con ésta, sin embargo, aquellas empresas que quieran incorporarlo para fines privados deben comprar a MySQL AB una licencia específica que les permita este uso.

MySQL es muy utilizado en plataformas Linux/Windows-Apache-MySQL-PHP/Perl/Python. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece como combinación con MySQL, ya que gracias a su rapidez es el gestor ideal para este tipo de aplicaciones. A continuación se listan algunas de sus características más destacadas:

- Soporte a multiplataforma
- Un amplio subconjunto de ANSI SQL 99, y varias extensiones
- Procedimientos almacenados
- Cursores
- Disparadores ("trigger")
- Soporte a Varchar
- Vistas actualizables
- Motores de almacenamiento independientes (MyISAM para lecturas rápidas, InnoDB para transacciones e integridad referencial)
- Soporte para SSL
- Permite sentencias SELECT anidadas
- Soporte completo para Unicode

En el paquete de aplicaciones elegido para desarrollar el proyecto, se incluye el módulo phpMyadmin. Este módulo es una herramienta escrita en PHP que actúa de interfaz con el sistema gestor de bases de datos a través de aplicaciones web, que permite ejecutar sentencias SQL en el navegador web y realizar operaciones sobre bases de datos, administración, tablas e importación y exportación de datos.

Durante todo el desarrollo del proyecto se ha utilizado esta aplicación para interactuar con la base de datos. En la siguiente figura se muestra la apariencia de la herramienta con la base de datos del proyecto en pantalla.

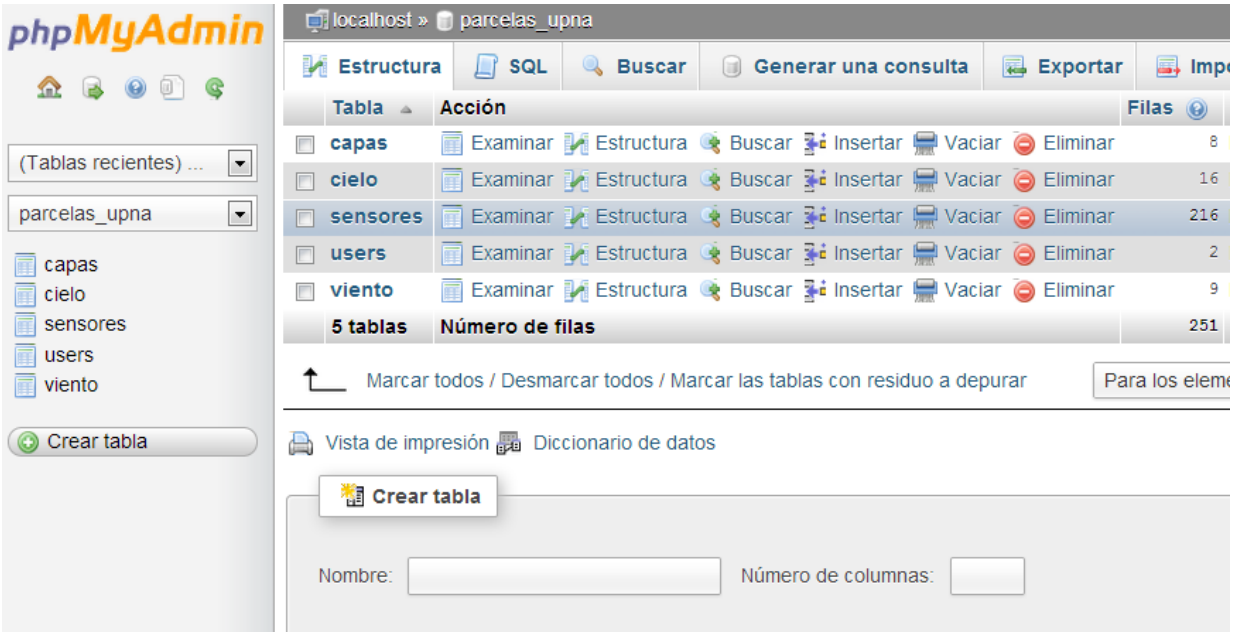


Ilustración 33: phpMyAdmin

Este paquete de aplicaciones ha sido elegido por ser el más adecuado y adaptarse a las necesidades planteadas. Este servidor contiene todas las herramientas necesarias para la realización de las tareas que comprenden este proyecto.

3.1.8. Criptografía SSL

SSL (Secure Socket Layers es un proceso) que administra la seguridad de las transacciones que realizan a través de Internet. El estándar SSl fue desarrollado por *Netscape*, junto con *Mastercard*, *Bank of America*, *MCI* y *Silicon Graphics*. Se basa en un proceso de cifrado de clave pública que garantiza la seguridad de los datos que se envían a través de Internet. Su principio consiste en el establecimiento de un canal de comunicación seguro (cifrado) entre dos equipos (el cliente y el servidor) después de una fase de autentificación.

El sistema SSL es independiente del protocolo utilizado; esto significa que puede asegurar transacciones realizadas en la Web a través del protocolo HTTP y también conexiones a través de los protocolos FTP, POP e IMAP. SSL actúa como una capa adicional que permite garantizar la seguridad de los datos y que se ubica entre la capa de la aplicación y la capa de transporte (por ejemplo, el protocolo TCP).

De esta forma, SSL es transparente para el usuario (es decir, el usuario puede no conocer que está usando SSL). Por ejemplo, el usuario que utiliza un navegador de Internet para conectarse a una página Web de comercio electrónico protegido por SSL enviará datos cifrados sin tener que realizar ninguna operación especial. Actualmente, casi todos los navegadores soportan el protocolo SSl.

Un servidor de Web seguro tiene una dirección URL que empieza con *https://*, en el que la "s" obviamente significa *secured*, *seguro*.

¿Cómo funciona SSL?

La seguridad de las transacciones a través de SSL se basa en el intercambio de calves entre un cliente y un servidor. Una transacción segura SSL se realiza de acuerdo al siguiente modelo:

- Primero, el cliente se conecta al servidor comercial protegido por SSL y pide la autentificación. El cliente también envía la lista de los criptosistemas que soporta, clasificada en orden descendente por la longitud de la clave.
- El servidor que recibe al solicitud envía un certificado al cliente que contiene la clave pública del servidor firmado por una entidad de certificación (CA), y también el nombre del criptosistema que está más alto en la lista de compatibilidades (la longitud de la clave de cifrado, 40 o 128 bits, será la del criptosistema compartido que tiene el tamaño de clave de mayor longitud).
- El cliente verifica la validez del certificado (y por consiguiente, la autenticidad del vendedor), luego crea una clave secreta al azar (más precisamente un supuesto bloque aleatorio), cifra esta clave con la clave pública del servidor y envía el resultado del servidor (clave se sesión).
- El servidor es capaz de descifrar la clave de sesión con su clave privada. De esta manera, hay dos entidades que comparten una clave que sólo ellos conocen. Las transacciones restantes pueden realizarse utilizando la clave de sesión, garantizando la integridad y la confidencialidad de los datos que se intercambian.

SSL en el proyecto

El servidor utilizado para este proyecto es gratuito y no dispone de encriptado SSL para proteger los datos. Para el desarrollo de este proyecto es suficiente el servidor del que se dispone, pero en caso de que una empresa usara este servicio, sería aconsejable que comprara un servidor con esta seguridad, que supone un coste muy bajo al mes y evita posibles intrusiones en el sistema.

3.2. Base de datos

Para tratar y explicar la implementación dela base de datos propuesta hay que referirse al lenguaje SQL, éste ha sido el nexo entre el diseño lógico realizado y el sistema gestor de base de datos.

Para definir la estructura de la base de datos se ha utilizado el lenguaje de definición de datos (LDD) y para el manejo de la estructura el lenguaje de manipulación de datos (LMD). En este punto nos vamos a centrar en los pasos seguidos para la implantación y las principales sentencias utilizadas para especificar la estructura delos datos, por tanto nos centraremos en sentencias del lenguaje LDD.

3.2.1. Crear objetos

Partiendo del diseño lógico, el primer paso es crear las tablas en concordancia. Para ello se ha utilizado la sentencia “CREATE TABLE”.

CREATE TABLE ”NOMBRE_TABLA” (
‘CAMPO_1’ tipo,
‘CAMPO_2’ tipo)

Creación de las tablas

Tabla 'capas'

```
CREATE TABLE capas
(
    fecha   date    not null,
    name   varchar(30)    not null,
    CONSTRAINT pk_nam PRIMARY KEY (name)
);
```

Tabla 'cielo'

```
CREATE TABLE cielo
(
    codigo          varchar(4)      not null,
    imagen          varchar(100)   not null,
    CONSTRAINT pk_codci PRIMARY KEY (codigo)
);
```

Tabla 'sensores'

```
CREATE TABLE sensores
(
    date            date            not null,
    name            varchar(32)     not null,
    value1          varchar(15)     not null,
    CONSTRAINT pk_sens PRIMARY KEY (date, name)
);
```

Tabla 'users'

```
CREATE TABLE users
(
    name            varchar(10)     not null,
    password        varchar(32)     not null,
    CONSTRAINT pk_user PRIMARY KEY (name)
);
```

Tabla 'viento'

```
CREATE TABLE viento
(
    codigo          varchar(2)      not null,
    imagen          varchar(100)    not null,
    CONSTRAINT pk_codvi PRIMARY KEY (codigo)
);
```

3.2.2. Consultas

Dar de alta un usuario

```
$consulta="INSERT INTO users(name,password) VALUES ('".$nom."','".$pas2."')";
$resultado=mysql_query($consulta);
```

Esta es la consulta realizada por el usuario Administrador para dar de alta un nuevo usuario invitado. Como se puede observar, se altera la tabla de usuarios realizando una inserción. Los valores a introducir se pasan con variables escapadas para maximizar la seguridad y no pasarlas directamente.

Coger valores en un intervalo temporal para un sensor

```
$consulta="SELECT date,value1 FROM sensores WHERE (name='parcela1sensor1')
AND (date BETWEEN '".$desde2."' AND '".$hasta2."') ORDER BY date ASC";
```

A continuación vemos la consulta más realizada en la web. Esta consulta es la que se usa para sacar los valores de un determinado sensor en un determinado intervalo temporal. Se seleccionan todos los valores y fechas para después poder mostrarlas en las nuevas gráficas. Siempre se le pasará el nombre del sensor que interesa conocer sus datos, y seguido los valores de las fechas limítrofes, escapados como antes. Por último todos los datos han sido ordenados para que en la gráfica aparezcan en orden, de la primera medición a la última. Esta consulta vale tanto para la gráfica de barras como para la de sectores.

Coger valores en un intervalo temporal para todos los sensores

```
$consulta="SELECT name,sum(value1) FROM sensores
WHERE date BETWEEN '".$desde2."' AND '".$hasta2."' GROUP BY name";
```

En el caso de la comparación de datos entre todas las gráficas, no se le pasa el nombre del sensor y sacará los datos de todos los sensores, con su cómputo global, que se consigue con la orden *'sum()'* en la consulta. Los intervalos de tiempo se pasarán igual que para un sensor sólo, y al final los datos se agruparán por nombre, para que realice las sumas de los datos de forma correcta.

Comprobar autentificación usuarios

```
$result = mysql_query(sprintf("select * from users where name= '%s' and
password = '%s'",mysql_real_escape_string($usuario),mysql_real_escape_string($password)));
```

El código anterior pertenece a la primera consulta que se realiza en el sistema, la autentificación de un usuario. El usuario y la password se pasan escapados a la consulta, para el tema de seguridad, puesto que son datos delicados que no se quiere que se saquen.

Insertar nueva capa KML

```
$consulta="INSERT INTO capas(fecha,name) VALUES (NOW(),'".$capa."')";
```

A la hora de insertar una capa, además del apartado referente a ftp, el sistema realiza una inserción a la tabla de datos de capas. Inserta el nombre de la capa, y como fecha de inserción toma la fecha actual con el comando '*NOW()*'.

Eliminar capas KML

```
$query="DELETE FROM capas WHERE name='".$valores[$i]."'";
```

Esta consulta se realiza dentro de un bucle, por eso la variable con los nombres de las capas a eliminar es un array, irá eliminando las capas de una en una mientras haya datos en el array. Se compone de una orden de DELETE sobre la tabla capas.

Dar de baja usuarios invitados

```
$query="DELETE FROM users WHERE name='".$valores[$i]."'";
```

Igualmente a la consulta anterior, la eliminación de usuarios se realiza dentro de un bucle al que se pasan los nombres de los usuarios a eliminar en las distintas posiciones de un array

3.3. Aplicación web

La aplicación web está implementada como la mayoría de las páginas, principalmente con lenguaje HTML y la hoja de estilos CSS. Para aumentar las funcionalidades de la aplicación se ha utilizado lenguaje PHP y JavaScript, que como queda reflejada en la información de dichos lenguajes, aportan dinamismo y personalización. A continuación vamos a describir la visión general de la página y a profundizar en los métodos llevados a cabo para implementar los aspectos más relevantes:

Visión general

En un primer vistazo el usuario accede a una página de inicio, algo general en multitud de páginas web. De una forma discreta y elegante se le ofrece al usuario un sistema de login mediante el cual dar paso a la aplicación en su totalidad. Dicho formulario de login cuenta con varios sistemas de seguridad y requisitos, el usuario a de introducir obligatoriamente los dos campos necesarios o el sistema le echará para atrás.

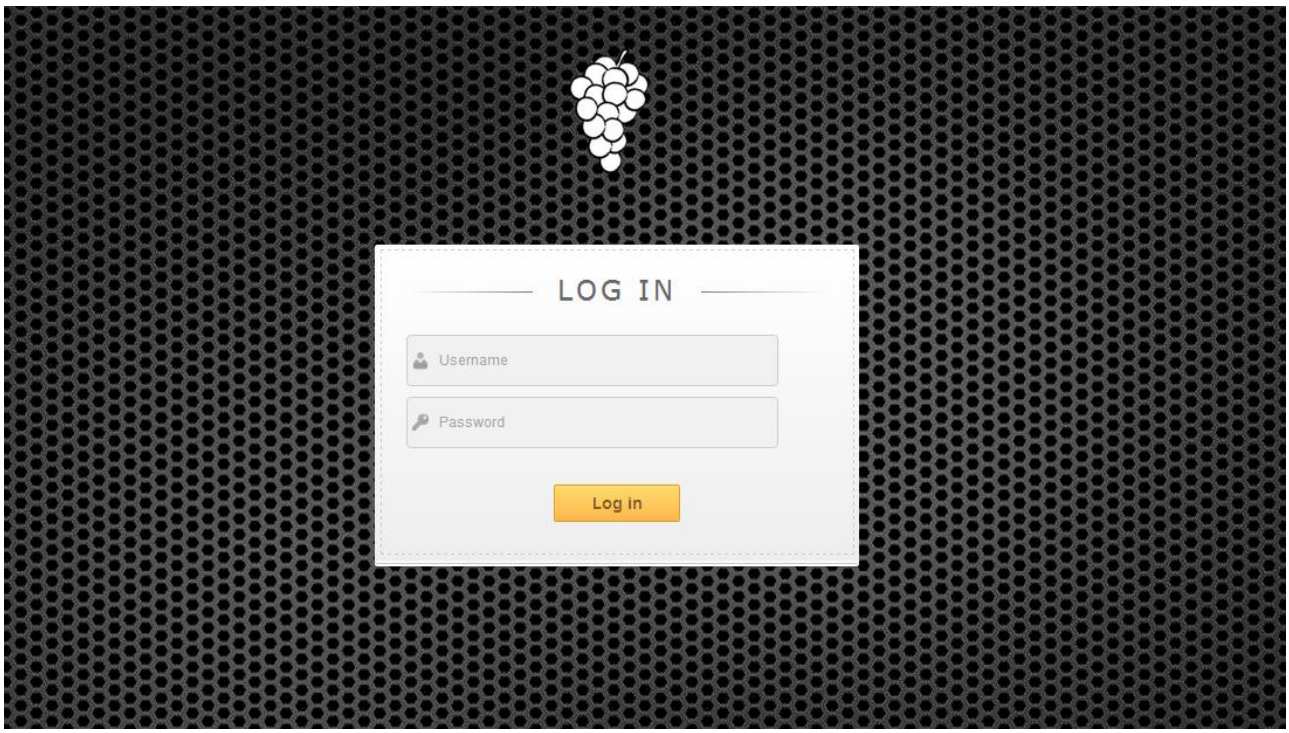


Ilustración 34: Login de autentificación

En la siguiente imagen vemos como la página principal, el index, consiste en poco más que el formulario mencionado, en el cual se llama a sesion.php que es donde se certifica la validez del usuario. Al tener todo en el mismo fichero php nos cercioramos que nadie puede acceder al código fuente y que quede en segundo plano el método usado para acceder a la web.

```
<form id="login" action="sesion.php" method="post">
        <h1>Log In</h1>
        <fieldset id="inputs">
                <input id="username" name="username" type="text" placeholder="Username"
                <input id="password" name="password" type="password" placeholder="Password"
        </fieldset>
        <fieldset id="actions">
                <input type="submit" id="submit" value="Log in">
        </fieldset>

</form>
```

Además se ha evitado la inyección SQL en dichos campos escapando el valor al realizar la sentencia SQL. Una vez introducidos los datos, se cifrará la contraseña y se compararán los campos con los de una tabla destinada a los usuarios en la base de datos propia del servidor; con este cifrado impediremos que un posible atacante de nuestro entorno web consiga dicha información.

```
if(isset($_POST['username']) && ($_POST['password'])){


                $usuario=$_POST['username'];
                $password=$_POST['password'];
                $password=md5($password);
```

Aquí podemos ver cómo lo primero que hace el archivo de autentificación es comprobar que le llega algo en ambos campos por el método post, y seguido lo guarda en variables y cifra la contraseña a md5, que es como están almacenadas las contraseñas en la base de datos.

La siguiente parte de la autentificación es conectar a la base de datos con la función mysql_connect y realizar una consulta a la tabla donde están almacenados los usuarios para ver si coincide lo que se ha pasado por el formulario de login con alguna fila de la tabla. Dichos datos no se pasan directamente a la consulta, sino que se indica que se le pasa un string y seguidamente las variables van escapadas con la función mysql_real_escape_string. Si el resultado de la consulta devuelve un valor de filas distinto de 0 es que la autentificación es correcta, se iniciará sesión y el fichero sql.php preparará la página principal de la web.

```
$connect=mysql_connect($host,$user,$pwd);
mysql_select_db($dbname)or die(mysql_error());

$result = mysql_query(sprintf("select * from users where name= '%s' and password =
'%s'",mysql_real_escape_string($usuario),mysql_real_escape_string($password)));


        if (mysql_num_rows($result) != 0) {

                $sesion=True;
                $user=$usuario;
                $_SESSION['sesion']=$sesion;
                $_SESSION['user']=$user;
                Header("Location: sql.php");
```

Además de todo lo relacionado con el login, a esta página de bienvenida se le ha añadido un logo propio realizado con Photoshop, algo característico de las viñas como es un racimo de uvas.

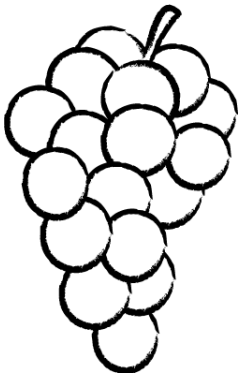


Ilustración 35: Logo aplicación

Una vez contrastada la información del login y dada por buena se accede a la interfaz principal desde la cual se podrá acceder al total de las características disponibles y datos de los cuales dispone la aplicación. Esta interfaz se estructura de forma sencilla para el usuario final, dejando claro todos los campos. Principalmente se compone de tres apartados claramente diferenciados.

Ilustración 36: Visión general interfaz

En la parte superior de la interfaz se distingue un menú con las diferentes opciones a visualizar. Diferenciaremos las partes configurables y tratables por el usuario y otra con la que comparar los datos propios con los recogidos por medios externos. Comenzaremos tratando la parte “usuario”.

Primeramente un apartado 'Home' que llevará al usuario siempre que este la solicite a la interfaz principal, o también dicho, al inicio. El siguiente punto del menú es 'Marca' que servirá para que el usuario haga marcas concretas en el mapa con sus correspondientes anotaciones y después pueda crear un fichero con esa marca y las pueda utilizar con posterioridad. Otro punto interesante es la opción 'Compara!', en este apartado el usuario accederá a una visualización gráfica sectorial en la cual podrá comparar los datos recogidos por los sensores, editando los sensores a comparar y fechas. A continuación podremos visualizar las marcas realizadas en el apartado anterior así como otras capas configuradas, esto es posible gracias a un apartado ('Sube/elimina capas'), aquí nos dará la opción de configurar las capas a mostrar, tanto subiendo al servidor nuevas capas o marcas, como eliminándolas de la lista. El último

punto de configuración propia es 'Qr', en este apartado se podrán generar códigos Qr tan utilizados hoy en día, y guardarlos posteriormente, podrán ser de varios tipos para los diversos usos del usuario, tanto geolocalización, como mensajes de texto, de e-mail, teléfonos, URL's.La parte restante del menú corresponde a fuentes externas con las cuales el usuario interactúa. El usuario administrador además contará con un apartado de gestión de sesiones de usuarios invitados para dar de alta/baja usuarios.

La primera fuente de datos es el apartado 'Aemet', dicho apartado mostrará previsiones semanales de cómo serán datos que interesan al usuario, para tener una "idea" y estar atento ante posibles imprevistos. La otra fuente de la que se nutre la aplicación es 'Meteo' (Navarra), estos datos ya no son previsiones sino que son datos reales recogidos por medios cien por cien seguros, los cuales son de gran utilidad para el usuario, puesto que con ellos puede comparar los recogidos por sus sensores con los recogidos por otros especializados de la zona, y poder optimizar sus rendimientos.

Continuando con los apartados de la interfaz, nos fijamos en el apartado de la parte izquierda, aparentemente sencillo pero no menos importante. Inicialmente, dicho apartado está compuesto por las capas y marcas configuradas por el usuario para su visualización. Para que dichas capas aparezcan hay que destacar que archivo antes nombrado, sql.php, prepara la lista a mostrar.

```
$consulta1="SELECT name FROM capas";
$resultado1=mysql_query($consulta1);

$i=0;
while ($fila = mysql_fetch_row($resultado1))
{
        $array[$i]=$fila[0];
        $i=$i+1;


}


$compactada=serialize($array);
$compactada=urlencode($compactada);
```

Después de autentificar el login lo que realiza el código es otra consulta a la base de datos, esta vez a la tabla de capas, seleccionándolas todas y almacenándolas en un array, y tras esto serializa el array a un flujo de bytes que seguidamente los transforma en una cadena codificada, que es lo que se pasa por la url finalmente. Con todo esto se aumenta la seguridad, evitando inyecciones en la url, ya que es cambiante constantemente.

```
header("Location: http://parcelas.x10.mx/portada.php?arr=".$compactada);
```

Apartados de no mucho tamaño como la configuración de las capas (sube/elimina), la gestión de usuarios por parte del administrador, o los distintos avisos al usuario, también se refrescarán en dicho apartado, haciendo la visualización y usabilidad lo más dinámica posible.

El último apartado de los que se compone la interfaz de la aplicación es el más visual y el que mayor tamaño necesita. Principalmente se utiliza para visualizar sobre el mapa las capas y marcas de las que dispone el usuario, por ello aparece un mapa inicialmente junto con las capas del apartado contiguo. Para poder visualizar el mapa en la web primeramente hay que enlazar en el código fuente el fichero JavaScript que se necesita para ejecutar el mapa, dicho código es proporcionado directamente por Google. El código se debe insertar en el archivo web en la que se desea mostrar el mapa, en este caso con extensión *.php, seguidamente de la cabecera <head> y con una etiqueta <script>. Anteriormente hacía falta logearse con tu cuenta gmail y que Google te proporcionara una clave, con esto controlaban el uso de su software, pero actualmente el uso de googlemaps está tan estandarizado que ya no es necesario.

```
<script type="text/javascript"
    src="http://maps.googleapis.com/maps/api/js?sensor=false">
</script>
```

Lo siguiente es introducir el código con las distintas opciones para cargar el mapa en la aplicación y posicionarlo en la zona deseada. En mi caso me interesa centrar el mapa en la zona donde se disponen las parcelas e indicarle un zoom para evitar el hecho de buscar la zona cada vez que se accede a la aplicación. Además, me ha parecido más útil trabajar con el mapa Satellite de Google, porque es una vista más real. Para facilitar la representación del mapa en la web, se le indica a la función que lo representará que debe crear dicho mapa en el div 'mapa', cuyas propiedades están definidas en la hoja de estilos CSS.

```
function initialize() {
    var layers = document.getElementsByTagName('input');
    var options = {
        center: new G.LatLng(42.4815,-1.65164),
        zoom: 15,
        mapTypeId: G.MapTypeId.SATELLITE,
        scaleControl: true,
        overviewMapControl: true,
        mapTypeControlOptions: {
            style:G.MapTypeControlStyle.DROPDOWN_MENU }
    };

    map = new G.Map(document.getElementById('mapa'), options);
```

Además de esta funcionalidad también se podrán visualizar otras que necesitan de espacio para mapas, códigos, datos... es el caso de los puntos del menú 'Marca', 'Aemet', 'Meteo' y 'Qr'. Para visualizar en esta sección los distintos puntos del menú superior de la página primeramente opté por redirigir toda la página a otra con la nueva información, pero además de esta información también tenía que repetir todo el formato de la página, por lo que no era una opción eficaz. Por estos motivos decidí refrescar solamente la parte principal donde se encuentra el mapa, sin tener que refrescar toda la página, esto aporta mucha velocidad y fluidez a la interfaz. Esta solución finalmente la he desarrollado usando iframes, de dos maneras distintas pero cuya estructura es similar.

```
<li><a href="Javascript: cargamarca();">Marca</a></li>

cargamarca=function(){
        var ma=document.getElementById("mapa");
        ma.innerHTML="<iframe id=\"iframe2\" src=\"http://www.parcelas.x10.mx/generakml.php\"
        marginheight=\"80px\"></iframe>"
}
```

El código superior muestra la primera alternativa usada. El apartado 'Marca' del menú de la página hace referencia a una función JavaScript descrita en la parte superior del código con los demás scripts. La función en sí indica que el div donde va a actuar es el div "mapa" y a continuación crea un iframe sobre ese div y ahí ya se puede refrescar únicamente el archivo que se desee, en el caso de la imagen refrescará el archivo "generakml.php".

```
<li><a href="Javascript: cargar('#mapa','aemet3.php');">Aemet</a></li>

function cargar(div, destino){
        $(div).load(destino);
}
```

Ahora podemos observar la segunda opción utilizada. En ella la opción del menú también llama a una función JavaScript, pasándole dos variables, en la primera se indica el div sobre el cual se desea refrescar, en nuestro caso otra vez el div "mapa" que es el de mayor tamaño y donde mejor se puede trabajar, y la segunda variable indica el archivo a mostrar.

Visto por encima el plano general de lo que sería la aplicación pasamos a profundizar cada uno de los apartados.

Home

Una de las principales funcionalidades para las que se ha diseñado este proyecto es para que el usuario tenga control visual de sus cultivos, sensores, y otros datos de su interés en todo momento y en cualquier lugar. Esto es lo que principalmente ofrece el 'Home' de la página web. El usuario de forma inmediata accederá a este apartado una vez pase con éxito el formulario de login, así a primera vista y de forma sencilla podrá controlar dónde se sitúan sus dominios e interactuar con los controladores y reguladores de éstos.

Ilustración 37: Apartado Home

La parte izquierda muestra de forma listada las diferentes capas y marcas configuradas a gusto del usuario. Estas capas mediante el uso de los seleccionables “chekbox” serán visualizadas en el mapa de su parte derecha. Estas capas o marcas son en realidad archivos .kml almacenados en el servidor de la página, al hacer click uno de ellos se ejecuta una función JavaScript que lo que hace es interpretar dicho archivo kml y en el caso de que no esté visualizado en el momento de clicar volcará la información

del archivo de forma visual en el mapa. Este hecho es posible gracias a que el Api de google.maps integra formas de parsear archivos *.kml, *.shp y otros muchos.

Ahora vamos a ver cómo se ha creado la lista de capas que el usuario puede visualizar y cómo hace para que se ilustren las capas que meramente son código en el mapa principal. Lo primero es coger los nombres de las capas que te llegan por las variables de la url. Estas variables se guardan en un array y seguido por cada elemento del array se crea un elemento input hasta que se termine el array.

```
            if ($mi_array!=NULL){
                    foreach ($mi_array AS $nombre){

    ?>
                                <input id="<?php echo $nombre;?>" rel="d4"/>
    <?php
                                echo $nombre."</br>";
                    }
            }
    ?>
```

Seguido de crear los inputs estos son tratados en la función initialize() de la que ya había hablado con anterioridad para crear el mapa. En esta función se almacenan todos los inputs en una variable 'layers' y al final de la función se trata cada elemento de 'layers' convirtiéndolos en elementos del tipo checkbox, y asignándoles su click a la función toggle() encargada de mostrar en el mapa el contenido de los archivos kml.

```
var layers = document.getElementsByTagName('input');

for (var i=0; i<layers.length; i++) {
    layers[i].type = 'checkbox';
    G.event.addDomListener(layers[i], 'click', toggle);
};
```

Entrando en análisis de la función toggle(), hay que aclarar que la primera vez que se ejecuta los elementos kml no han sido tratados, por ellos crea un objeto por cada uno de los kml que se encuentran en el servidor y los asocia al mapa, el modo de visualización lo pone a false porque únicamente los está inicializando. Una vez que el usuario haga click sobre uno de los checkbox creados, ejecutará de nuevo la función pero como los objetos ya han sido inicializados, irá al apartado de la función de visualización. Para mostrarlos sobre el mapa tiene que estar "displayIsOn=false".

Seguido al clicar dibujará el contenido del kml en el mapa y pondrá “displayIsOn=true” para que cuando se quiera quitar la visualización del mapa haga justo lo contrario.

```
function toggle() {
            if (!this.kmlLayer ) {
                      this.kmlLayer = new G.KmlLayer(
                              'http://www.parcelas.x10.mx/' + this.id + '.kml',
                              { preserveViewport:true } );
                      displayIsOn = false;

            }
            if ( this.displayIsOn ) {

                      this.kmlLayer.setMap( null );
                      this.displayIsOn = false;
            }
            else {

                      this.kmlLayer.setMap( map );
                      this.displayIsOn = true;

            };

    };
```

Las capas a destacar en el caso del este usuario son principalmente aquellas en las cuales tiene recogidas sus parcelas, de tal modo que al visualizarlas obtiene sobre el mapa el contorno de ellas, ahorrándose el trabajo de buscarlas por sí mismo en el mapa cada vez. Además de éstas, otras muy importantes son en las cuales aparecen los sensores, protagonistas de la aplicación, puesto que serán los que tengan los datos que podrá ver el usuario de las distintas zonas de sus cultivos. A través de los iconos que quedarán plasmados en el mapa con la ubicación de los sensores, el usuario tiene las opciones de poder ver los datos recogidos tanto en forma de barras como en forma de

sectores. Hay que decir que los datos que se visualizarán serán únicamente de un sensor, de aquel que el usuario haya escogido en el mapa.

Ilustración 39: Visualización de capas en el mapa

Gráficas

Una vez el usuario elige la forma en la que desea ver los datos, se le abrirá en otra ventana la gráfica resultante; esta forma de visualizar los datos es de gran utilidad y está pensada para que el usuario trabaje a doble pantalla, de tal forma que en una pantalla tendrá la interfaz principal de la aplicación con el mapa y las diferentes opciones, y en la otra pantalla la gráfica seleccionada en un instante concreto, de un vistazo tendrá una visión global y de forma rápida alternará la información a mostrar en la gráfica.

Para distinguir entre las distintas parcelas y sensores, dentro del kml de las parcelas las diferenciaremos en los href que redirigen al apartado web de gráficas de este modo.

```
<a href="http://www.parcelas.x10.mx/grafica_parcela1sensor1.php?par=p1s1">Barras</a>
<a href="http://www.parcelas.x10.mx/grafica_sectores.php?par=p1s1">Sectores</a>
```

Como podemos observar basta con distinguir la parcela y el sensor en la variable ‘par’, en el caso de tratar los datos del sensor número 1 de la parcela 1 lo denotaremos ‘p1s1’, para otros casos será parecido, ‘p4s1’, ‘p4s3’… Esta variable se pasa por url a uno de los archivos encargados de las gráficas, o al fichero de gráficas de barras o al fichero de gráficas de sectores. En ambos ficheros lo primero que se hace es recoger con el método GET el contenido de la variable ‘par’ y almacenarlo en una variable. Una vez conocido qué datos se quieren para construir la gráfica, se procede a hacer una consulta sql a la tabla “sensores” donde están los datos y se almacenan en dos arrays, uno de fechas y otro de valores.

Un inconveniente que he encontrado a la hora de hacer las gráficas es que a la hora de plasmar los datos se plasman desde JavaScript y los datos que obtengo de la sentencia SQL están en variables php, y hay que transformarlas. Para ello, dentro de un script del código fuente, creo dos arrays JavaScript y los relleno con código php, quedando algo así.

```
<script type="text/javascript">
        var jsarray      = [];
        var jsarray2      = [];

                <?php

                    for($i=count($array)-10; $i < count($array); $i++ ){

                        echo 'jsarray['.$i.'] = "'.$array[$i].'";';

                    }

                    for($j=count($array2)-10; $j < count($array2); $j++ ){

                        echo 'jsarray2['.$j.'] = "'.$array2[$j].'";';

                    }


                ?>

</script>
```

De los datos que tenía en los arrays php sólo me interesa los 10 últimos de cada uno, porque en esta primera visualización se mostrarán las últimas muestras, para que el usuario pueda ver de un simple vistazo cómo han ido las estadísticas en el último espacio temporal. Para disponer los datos del array en la gráfica se utiliza una función JavaScript que se carga seguido en el body, <body onload=”grafica()”>

Las siguientes ilustraciones de código nos van a servir para analizar con más detalle el proceso de creación de las gráficas. La primera función es la correspondiente a la gráfica de barras. En ella podemos observar como lo primero que hace es crear un objeto array con el que poner los datos, una a una irá recorriendo las posiciones de los arrays creados anteriormente en JavaScript, distinguiendo por etiquetas diferenciadas en la gráfica, una para las fechas y otra para el valor total de la medición. Seguido se crea un objeto específico de la librería de la gráfica y sobre ese objeto se configuran apartados de la gráfica como los márgenes, ángulos, etiquetas, los ejes, el tipo de gráfico, el color y donde posicionar la gráfica, en un div creado en la parte html.

La segunda ilustración de función de gráfica pertenece a la gráfica de sectores. El mecanismo de la función es prácticamente igual al diagrama de barras. En lugar de tomar las últimas 10 mediciones, se han tomado 6 para que no haya muchas particiones en el sector. A parte de esto, podemos ver que el objeto que se crea es del tipo ‘AmPieChart’ que según la librería es el que corresponde a este tipo de gráfica. El resto de configuraciones se asemejan: etiquetas, color de fuente, div donde dibujarla, y las específicas de angulación del 3D.

```
function grafica() {

                var chartData=new Array();
                var i;
                for (i=(jsarray.length)-10;i<jsarray.length;i++){
                chartData.push({"fecha": jsarray2[i],"total": jsarray[i]});

                }


                var chart = new AmCharts.AmSerialChart();
                chart.dataProvider = chartData;
                chart.categoryField = "fecha";
                chart.marginTop = 15;
                chart.marginLeft = 55;
                chart.marginRight = 15;
                chart.marginBottom = 80;
                chart.angle = 30;
                chart.depth3D = 15;
                chart.color="white";

                var catAxis = chart.categoryAxis;
                catAxis.gridCount = chartData.length;
                catAxis.labelRotation = 30;

                var graph = new AmCharts.AmGraph();
                graph.balloonText = "[[category]]: [[value]]";
                graph.valueField = "total"
                graph.type = "column";
                graph.lineAlpha = 0;
                graph.fillAlphas = 0.8;
                graph.fillColors ="#EEF215";
                chart.addGraph(graph);

                chart.write('graficab');


        }



function grafica() {
        var chartData=new Array();
                var i;
                for (i=(jsarray.length)-6;i<jsarray.length;i++){
                chartData.push({"fecha": jsarray2[i],"total": jsarray[i]});


                }

                chart = new AmCharts.AmPieChart();
                chart.dataProvider = chartData;
                chart.titleField = "fecha";
                chart.valueField = "total";
                chart.outlineColor = "#FFFFFF";
                chart.outlineAlpha = 0.8;
                chart.outlineThickness = 2;
                // esto hace el chart 3D
                chart.depth3D = 15;
                chart.angle = 20;
                chart.color="#15F2E3";

                // WRITE
                chart.write("graficas");
}
```

En la pantalla de las gráficas, tanto la de barras como la sectores, el usuario dispondrá en la parte superior de un formulario con calendario desplegable para introducir las fechas entre las cuales desea saber los datos recogidos por los sensores. A continuación de la introducción del intervalo temporal, se obtiene en su parte central-inferior una amplia gráfica con datos que estén exclusivamente dentro del intervalo, anotando a un margen la fecha en la cual está recogida esa medición y el porcentaje que representa sobre el total del intervalo. El colorido es algo a destacar en las gráficas. La gráfica de barras dispone de una paleta gradual de colores a elegir, de esta forma, el usuario puede tener abierta más de una gráfica, con intervalos de tiempo idénticos, y puede diferenciar de forma más visual las diferencias entre las mediciones de sensores distintos y comparar más fácilmente. Por su parte, la gráfica de sectores sacará cada medición de un color distinto y así poder comparar mediciones de la misma fecha por colores. Además esta gráfica dispone de la opción de clicar sobre una medición para destacarla, de tal forma que sobresaldrá del círculo sobre las demás, pudiendo hacer esto con todas para tener más distancia a la hora de visualizar. Todos los datos representados en las gráficas están obtenidos de una base de datos, en la cual los propios sensores introducirán los valores recogidos.

Relativo a la representación de gráficas con datos en un determinado espacio temporal, se procede ahora a profundizar en la dinámica utilizada. Lo primero a destacar es el formulario utilizado en la parte superior, donde el usuario elige las fechas y el color (en el caso de las barras). Este formulario llama en el apartado 'action' del mismo a una función JavaScript, similar a lo que ocurría con varios apartados del menú.

```
cargagrafica=function(){
        var gr=document.getElementById("chartContainer");

         var fd = document.getElementById("fecha_desde").value;
         var fh = document.getElementById("fecha_hasta").value;

         var co =document.getElementById("colores").value;
        alert('Desde '+fd+' hasta '+fh);
        gr.innerHTML="<iframe id=\"iframegra\" src=\"http://www.parcelas.x10.mx/datossql.php?
        fd="+fd+"&fh="+fh+"&parjs="+parjs+"&co="+co+"\" marginwidth=\"80px\" marginheight=\"80px\">

}
```

Lo que hace esta función JavaScript, es comenzar recogiendo las variables que interesan para la futura gráfica a representar. En ambas gráficas se necesita recoger las "fecha_desde", la "fecha_hasta" y el tipo de parcela y sensor a la que pertenecen los datos ('parjs'), y en la de barras además de estas también se necesita el color en el que sea desea dibujar. Esto se consigue recogiendo en contenido de los inputs, como se indica en la función. Además de esto se necesita crear un iframe en el lugar donde aparece la primera gráfica, y una vez ahí pasar estas variables por url al fichero que representará la gráfica.

El último paso para crear las nuevas gráficas es que los ficheros encargados de estas gráficas actúen. Lo primero que hacen es corroborar que las fechas introducidas son válidas. Para este hecho, se realiza un 'explode' bastante usual en programación, para separar la fecha en día, mes, año. Seguido se comprueba que la segunda fecha es mayor que la primera y se prosigue. A continuación no queda más que realizar una nueva consulta a la base de datos y recoger los datos pertenecientes a la parcela y al sensor en las fechas indicadas, y volver a representar esos datos como se ha citado anteriormente.

La siguiente utilidad a profundizar es una herramienta que complementa a estas gráficas. Era más que aconsejable que los datos reflejados en las gráficas se pudieran descargar, para que el usuario disponga de los archivos por si necesita trabajar con ellos. El mejor formato es *.xls, un formato de Excel, para así clasificar los datos en filas y columnas, y que quede toda la información organizada. Para este proceso hay una librería llamada PHPExcel, y a continuación entraremos más en detalle sobre ella.

PHPExcel

PHPExcel es una librería php que ayuda a leer y escribir hojas de cálculo en diversos tipos de formatos, por lo que se puede trabajar prácticamente con todas las versiones de Excel que existen hoy en día. Los formatos que acepta esta librería son:

Datos de entradas

- Excel 2007 en adelante
- BIFF5 (Excel 5.0 / Excel 95)
- BIFF8 (Excel 97 y posteriores)
- PHPExcel Serialized Spreadsheet
- Symbolic Link (SYLK de Microsoft)
- CSV (Comma Separated Values)

Datos de salidas

- Excel 2007 en adelante
- BIFF8 (Excel 97 y posteriores)
- PHPExcel Serialized Spreadsheet
- PDF
- HTML
- CSV (Comma Separated Values)

Para poder utilizar PHPExcel, debes asegurarte que la versión de php de tu servidor sea 5.2 o superior. Lo siguiente es descargarte la librería y con esto ya estaría listo para poder utilizar la herramienta.

Otro punto a favor de esta librería es que en su propia página se dispone de multitud de ejemplo de todas las opciones que se pueden realizar con ella, así como documentación detallada, un foro de discusiones de todo tipo, consejos… Con todo esto vamos a ver cómo quedaría resuelto el problema de la descarga de los datos de los sensores.

```
creaexcel=function(){
        var fd = document.getElementById("fecha_desde").value;
        var fh = document.getElementById("fecha_hasta").value;
        window.location ="http://www.parcelas.x10.mx/excel.php?fd="+fd+"&fh="+fh+"&parjs="+parjs;
}
```

Primeramente, se ha colocado un botón con el logo de Excel en la página principal de las gráficas, porque necesitaba recoger las fechas y el tipo de parcela y sensor, y desde el iframe estos datos son inaccesibles. Entonces, una vez clicado el botón se ejecuta la función superior, la cual como en otras ocasiones recoge las fechas y seguido como no hay que actualizar nada sino descargar estos datos se pasan con 'window.location' al fichero encargado de crear y rellenar el archivo *.xls.

```
if ($registros > 0) {
        require_once './PHPExcel/PHPExcel.php';
        $objPHPExcel = new PHPExcel();
        $i=3;
        while ($registro= mysql_fetch_object($resultado)) {
                $objPHPExcel->setActiveSheetIndex(0)
                        ->setCellValue('A'.$i, $registro->date)
                        ->setCellValue('B'.$i, $registro->value1);
                $i++;
        }
        //Agregar encabezados
        $objPHPExcel->getActiveSheet()->SetCellValue('A1','Fecha');
        $objPHPExcel->getActiveSheet()->SetCellValue('B1','Valor');

}

$f= Date("Y-m-d_His");
header('Content-Type: application/vnd.ms-excel');
$fecha01="Reporte_".$f.".xls";
header('Content-Disposition: attachment;filename='.$fecha01.'');
header('Cache-Control: max-age=0');
$objPHPExcel->getActiveSheet()->setTitle('Reporte');
$objWriter=PHPExcel_IOFactory::createWriter($objPHPExcel,'Excel5');
$objWriter->save('php://output');
exit;

}
```

Al igual que los ficheros que ilustraban la gráfica, este fichero comprueba que las fechas son correctas y realiza la consulta a la base de datos. Si el número de filas que devuelve la consulta es mayor que cero entonces se puede crear el archivo Excel. Una vez se procede a pasar los datos de la consulta al nuevo fichero, lo primero que necesita es hacer referencia a un fichero clave en este proceso, 'PHPExcel.php', donde se encuentran las especificaciones de funciones, clases, operaciones, etc. Estas funciones están declaradas en otros ficheros a los que se hace referencia desde el anterior, el principal y el que organiza todo.

El siguiente paso es crear un objeto PHPExcel que es el que puede ejecutar las funciones. La primera función que realiza es ir rellenando las columnas, se le indica que serán las columnas A y B, a partir de la fila 3 en adelante. Una vez escritos los datos, añadimos los encabezados en la primera fila de dichas columnas. La última parte del proceso es puntualizar sobre el fichero, el nombre que será "Reporte_'Fecha'_'HoraMinutosSegundos'", para evitar repeticiones; el tipo de fichero de salida que será *.xls, el tipo de los datos de salida que será Excel5 y la opción 'save' de escritura para que se guarde como descarga.

Marca

La siguiente funcionalidad tiene como objetivo que el usuario de la aplicación pueda realizar de forma propia una anotación o marca sobre el mapa, sin necesidad de recurrir al programador de la aplicación para que le realice una configuración puntual.

Entrando un poco más en detalle, esta funcionalidad está dividida en tres apartados claramente diferenciables, uno para crear marcas, otro para múltiples líneas consecutivas y el último para crear polígonos y así delimitar zonas. Al igual que la interfaz principal, esta también dispone de un mapa con el que interactuar, donde el usuario podrá realizar las anotaciones oportunas. Además del mapa, dispone de un apartado superior en el cual se puede realizar nuevas marcas, deshacer marcas mal hechas u obtener el código fuente del archivo KML; a esto le complementa un buscador donde poder introducir una ubicación para que aparezca automáticamente en el mapa. En la parte izquierda del mapa, el usuario encuentra una caja de coordenadas donde irán apareciendo las coordenadas seleccionadas en el mapa.

Ilustración 40: Apartado de creación de marcas

Una vez el usuario ha realizado una marca visual en el mapa, dispone de un bocadillo de texto en el que poder especificar datos de la marca como el identificador o la descripción para cualquier tipo, o los colores y anchura de borde en el caso de polígonos y múltiples líneas consecutivas. Tras esto el usuario debe guardar, y seguido ya podrá guardar esa marca en un archivo *.kml, esto es posible ya que tanto las coordenadas, como las opciones configurables están almacenadas en variables, y al generar el código KML se ejecuta una función JavaScript que lo único que hace es completar una plantilla (de marca, poliLinea o polígono) con el contenido de las variables anteriormente completadas.

Ilustración 41: Creación de polígono KML

Cualquier usuario puede seguir los pasos para obtener un código KML válido para su futuro uso, dicho código no tiene más que ser pegado en un archivo con extensión *.kml. Únicamente el usuario Administrador podrá subir dichos archivos a la aplicación para su visualización, para evitar malas intenciones de usuarios invitados.

Esta función es muy útil porque el usuario puede realizar las marcas desde cualquier lugar, siempre que disponga de un dispositivo con conexión a internet, no será necesario llegar a la oficina o lugar de trabajo para hacerlo, en el momento que necesite lo podrá realizar, situación cotidiana para trabajadores del campo.

Compara!

El siguiente apartado del menú tiene como fin realizar una visión global de los datos recogidos a través de los sensores instalados en las parcelas. Antes el usuario había podido visualizar estos datos, en gráficos, pero a nivel individual. Ahora el usuario podrá contrastarlos con otros de diferentes áreas geográficas y diferentes bioclimas.

Al igual que las anteriores gráficas, esta también será estará diseñada para el entorno de doble pantalla. De esta manera el usuario puede estar mirando al mismo tiempo la interfaz principal de la aplicación con el mapa y las parcelas dibujadas en el mismo, o comprobando los datos recogidos por fuentes externas en una determinada fecha, y por otro parte en la pantalla adyacente comprobar los datos recogidos por todos los sensores, su respectiva comparativa y realizar un estudio con la suma de la información global.

La interfaz correspondiente a esta sección es muy similar a la utilizada para representar los datos de un sensor. En la parte superior posee un formulario con dos campos en los cuales introducir manualmente las fechas entre las que desea buscar los datos, o si lo prefiere, dispone dos calendarios desplegables donde elegirlas de forma visual. A continuación, en la parte central y principal de la sección, se sitúa la gráfica, de sectores, que hace apreciable de un solo vistazo los diferentes porcentajes obtenidos por cada sensor, cada sector es de un color distinto para simplificar y disponen la opción de separarse del círculo completo para resaltarse más. A su vez, si se desplaza el ratón por encima de cualquier sector aparecerá una nota resaltada del mismo color con la información correspondiente. La parte inferior corresponde a un menú de configuración, con un trasfondo basado en JavaScript, con el cual poder elegir entre dos marcos de opciones. En el primer marco situado justo por debajo de la gráfica se encuentra la lista con los sensores descritos en la gráfica, lo cual será de esta forma porque estarán en el resultado de la sentencia temporal realizada. Dentro de él, el usuario puede cancelar la visualización de los datos de un sensor ejerciendo clic

sobre el sensor o el color que lo identifica. El marco que ocupa la posición más baja controla el ámbito relacionado con el diseño gráfico; la etiqueta descriptiva del sensor tiene la propiedad de poder situarse o bien fuera del contorno del sector (como viene por defecto), o bien introducirla dentro del perímetro, jugando con esto ves los porcentajes dentro de la gráfica, facilitando su comprensión. La otra opción modificable y que es de recalcar es el cambio de la dimensiones de la gráfica, pudiendo pasar de la clásica visión 2D (más plana), a otra más amplia y con profundidad como es la 3D, esto ya depende del gusto del usuario y para el tipo de presentación que la pueda necesitar.

En un primer instante, el usuario visualizará los datos totales de los sensores, apareciendo por pantalla el cómputo total de cada sensor, con el porcentaje correspondiente sobre el total de los datos. Al igual que con las gráficas de cada sensor, al realizar una consulta temporal en la parte superior, se realiza una consulta sql obteniendo la suma de los datos de cada sensor en ese intervalo temporal. A continuación estos datos se visualizarán sobre un iframe, para no tener que refrescar toda la página.

El otro apartado semejante a las anteriores gráficas es la opción de poder descargar los datos en formato *.xls. Al igual que las otras gráficas se realiza este hecho mediante el fichero 'excel.php'. Dentro de este fichero la única diferencia con respecto a los datos es que en el caso de la gráfica para comparar, los datos van en relación al sensor y no a la fecha.

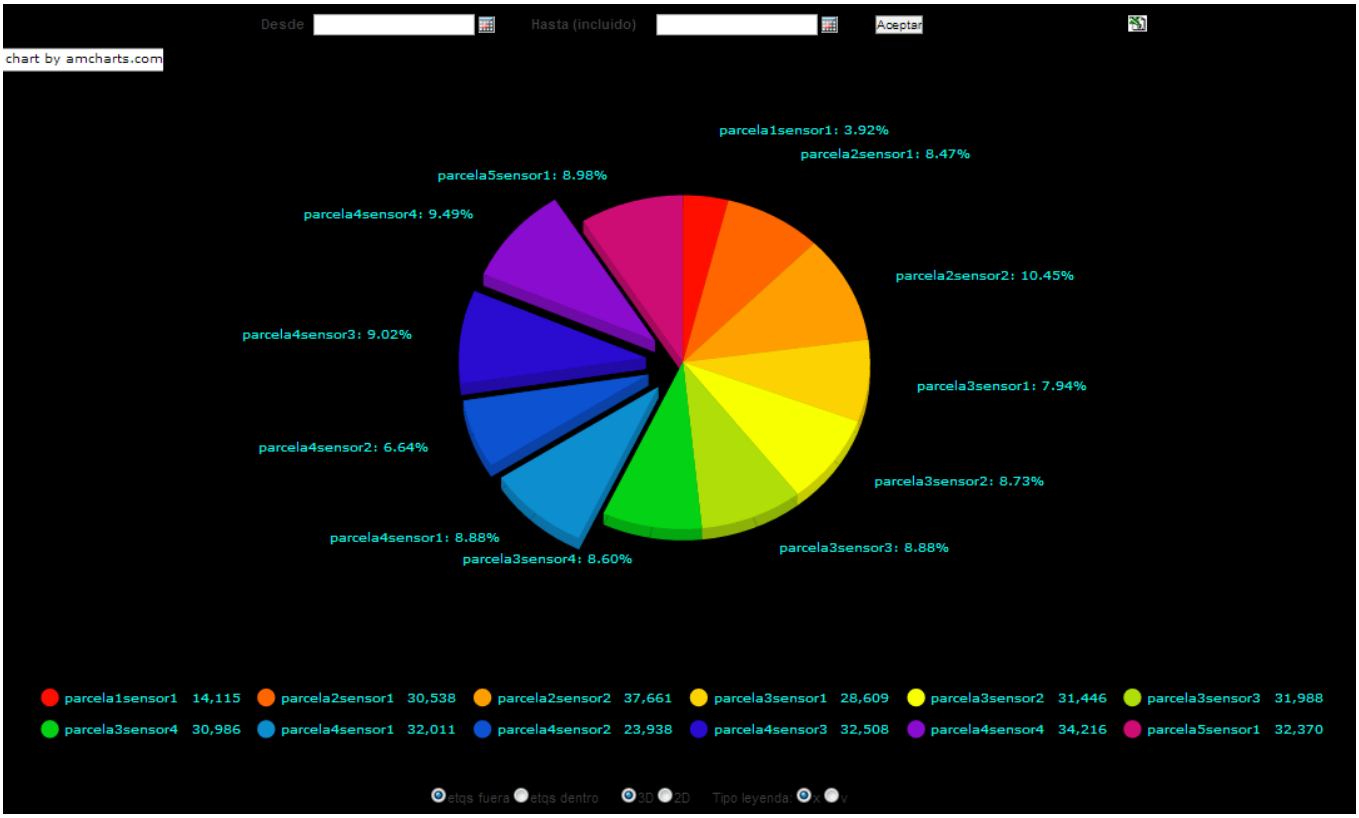


Ilustración 42 Comparación de los sensores en 3D

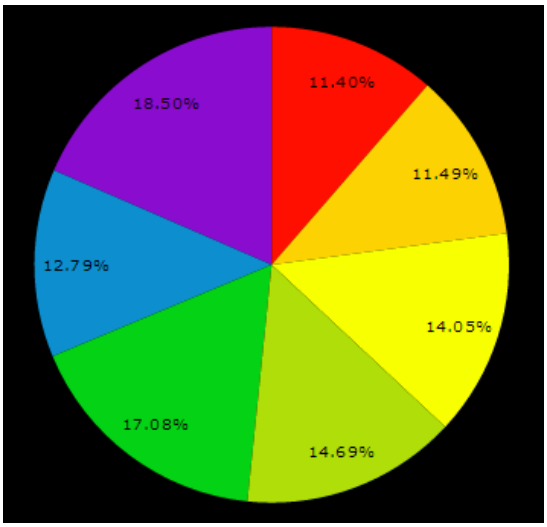


Ilustración 43: Visualización de datos en 2D, con las etiquetas dentro

Aemet

A continuación, entramos en un apartado meramente visual, sin ninguna interacción por parte del usuario. Hay que destacar que lo primero que hace la aplicación web nada más entrar al apartado home es lo siguiente:

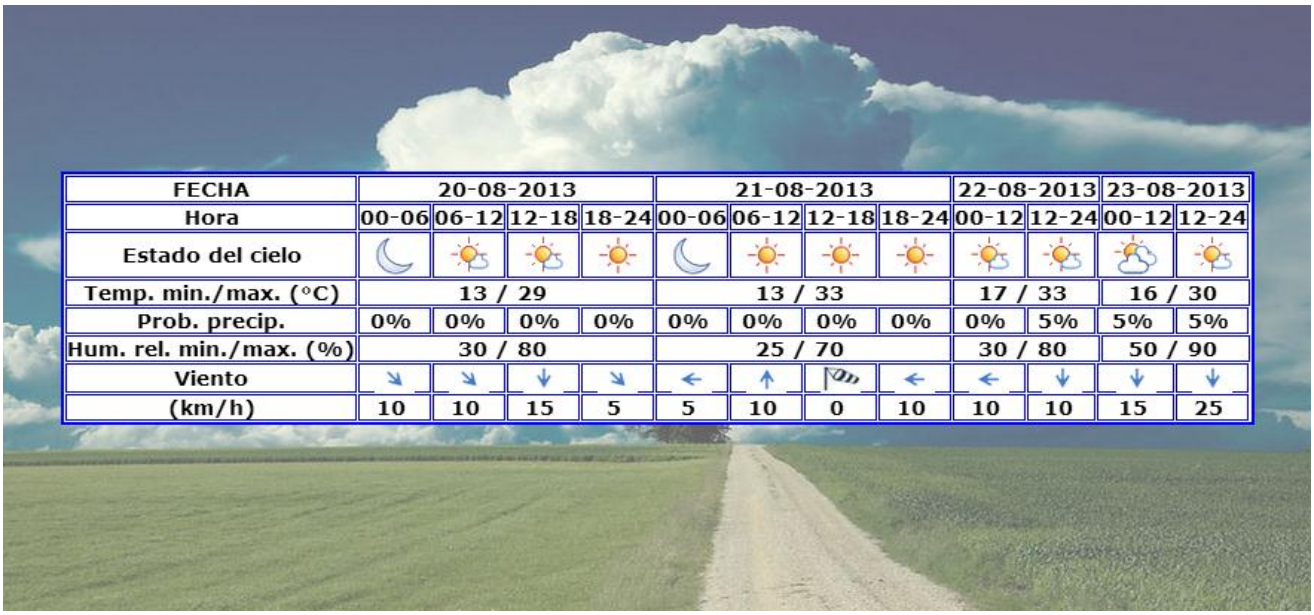
```
<?php
        $fp = fopen("ejemplo.xml","w");
        $url = "http://www.aemet.es/xml/municipios/localidad_31191.xml";
        $content = file_get_contents($url);
        fwrite($fp, $content);
        fclose($fp);

?>
```

Como podemos observar, se abre un archivo *.xml en modo escritura y se rellena con todo el contenido de la url que le sigue. Dicho archivo perteneciente a la url, corresponde con el archivo que la agencia de meteorología Aemet ofrece a la población, concretando la localidad con la que interesa para la zona donde se ubican la mayor parte de las parcelas. Dicho archivo se guarda en el directorio principal del servidor y tiene el siguiente aspecto

```
<?xml version="1.0" encoding="ISO-8859-15" ?>
<root id="31191" version="1.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http:
xsi:noNamespaceSchemaLocation="http://www.aemet.es/xsd/localidades.xsd">
        <origen>
                <productor>Agencia Estatal de Meteorología - AEMET. Gobierno de España</proc
                <web>http://www.aemet.es</web>
                <enlace>http://www.aemet.es/es/eltiempo/prediccion/municipios?l=31191</enlac
                <language>es</language>
                <copyright>&#169; AEMET. Autorizado el uso de la información y su reproducci
</copyright>
                <nota_legal>http://www.aemet.es/es/nota_legal</nota_legal>
        </origen>
        <elaborado>2013-08-20T09:05:03</elaborado>
        <nombre>Olite/Erriberri</nombre>
        <provincia>Navarra</provincia>
        <prediccion>
                <dia fecha="2013-08-20">
                        <prob_precipitacion periodo="00-12">0</prob_precipitacion>
                        <prob_precipitacion periodo="12-24">0</prob_precipitacion>
                        <prob_precipitacion periodo="00-06">0</prob_precipitacion>
                        <prob_precipitacion periodo="06-12">0</prob_precipitacion>
                        <prob_precipitacion periodo="12-18">0</prob_precipitacion>
                        <prob_precipitacion periodo="18-24">0</prob_precipitacion>
                        <cota_nieve_prov periodo="00-12"></cota_nieve_prov>
                        <cota_nieve_prov periodo="12-24"></cota_nieve_prov>
                        <cota_nieve_prov periodo="00-06"></cota_nieve_prov>
                        <cota_nieve_prov periodo="06-12"></cota_nieve_prov>
                        <cota_nieve_prov periodo="12-18"></cota_nieve_prov>
                        <cota_nieve_prov periodo="18-24"></cota_nieve_prov>
                        <estado_cielo periodo="00-12" descripcion="Poco nuboso">12</estado_c
                        <estado_cielo periodo="12-24" descripcion="Poco nuboso">12</estado_c
                        <estado_cielo periodo="00-06" descripcion="Despejado">11n</estado_ci
                        <estado_cielo periodo="06-12" descripcion="Poco nuboso">12</estado_c
                        <estado_cielo periodo="12-18" descripcion="Poco nuboso">12</estado_c
                        <estado_cielo periodo="18-24" descripcion="Despejado">11</estado_cie
                        <viento periodo="00-12">
                                <direccion>NO</direccion>
                                <velocidad>20</velocidad>
                        </viento>
```

Seguidamente, cuando el usuario accede al propio apartado, lo que principalmente ocurre es un parseo del archivo descargado para una visualización más sencilla para el usuario. Primeramente la opción elegida fue una visualización únicamente de texto, señalando los principales factores climáticos por día, pero con esto todavía costaba realizar una comparación temporal de un solo vistazo. La segunda opción que se ha contemplado es la realización de un widget, requiere una mayor complejidad de código y trabajo pero el resultado es el deseado. Ahora cualquier usuario, sin necesidad de ser entendido, ni capaz de realizar estudios comparativos, podrá conocer las previsiones climáticas y así poder anticiparse a ellas.



FECHA	20-08-2013				21-08-2013				22-08-2013		23-08-2013	
Hora	00-06	06-12	12-18	18-24	00-06	06-12	12-18	18-24	00-12	12-24	00-12	12-24
Estado del cielo												
Temp. min./max. (°C)	13 / 29				13 / 33				17 / 33		16 / 30	
Prob. precip.	0%	0%	0%	0%	0%	0%	0%	0%	0%	5%	5%	5%
Hum. rel. min./max. (%)	30 / 80				25 / 70				30 / 80		50 / 90	
Viento	↘	↘	↓	↘	←	↑		←	←	↓	↓	↓
(km/h)	10	10	15	5	5	10	0	10	10	10	15	25

Ilustración 44: Widget Aemet

Para comenzar con el parseo del archivo descargado desde Aemet lo primero que se debe hacer es cargarlo en php para poder trabajar su información.

```
$objDOM = new DOMDocument();
$objDOM->load("ejemplo.xml");
```

Seguidamente se ha procedido a distribuir los datos interesantes para el usuario en una tabla, fácil de entender. De todo el archivo, que abarca datos de una semana, únicamente se tratan los de los 4 días primeros, puesto que al ser predicciones, son los que mayor fiabilidad ofrecen. Los datos de los días, en lugar de cogerlos del archivo, con php también se puede sacarlos, a partir de la fecha actual hasta tres días más tarde.

```
<th>FECHA</th>
<?php
        for ($i=1;$i<=4;$i++){
        $cuenta=$i-1;
        $date2=date("d-m-Y",strtotime("+$cuenta day"));
?>
        <th colspan="4"><?php printf("%s",$date2); ?></th>
<?php
        }
?>
```

El resto de datos sí que necesitan ser sacados del archivo *.xml. Para ello en primer lugar, se extrae del objeto anteriormente creado de la clase *DOMDocument*, toda la información que haya dentro de cada etiqueta día (probabilidad de precipitaciones, temperaturas, viento…).

```
$prediccion = $objDOM->getElementsByTagName("dia");
```

A continuación se trata el interior de cada etiqueta día, que se encuentra en la variable '$prediccion'.

```
<th>Temp. min./max. (&#176;C)</th>
<?php
        foreach( $prediccion as $dia ){
                if($j<=3){
                        $j=$j+1;
                        $temp = $dia->getElementsByTagName("temperatura");
                        foreach( $temp as $cc )
                        {
                                $maxs = $cc->getElementsByTagName("maxima");
                                $mins = $cc->getElementsByTagName("minima");
                                $max = $maxs->item(0)->nodeValue;
                                $min = $mins->item(0)->nodeValue;
                        }
?>
<th colspan="4"><?php printf("%s",$min); ?> / <?php printf("%s",$max); ?> </th>
<?php
                }
        }
        $j=0;
?>
```

Este es un ejemplo más visual de cómo funciona el parseo de la información del fichero. Como se comentaba anteriormente, se trata cada predicción que contendrá la información de un día. En el caso del código superior, se desea sacar la temperatura, por ello se selecciona la información que está dentro del *tag* "temperatura". Seguido obtenemos la información que está los tags de interés, con el comando nodeValue, y por último quedaría imprimir el contenido de estas variables en su posición correcta de la tabla.

En los casos del estado del cielo y el viento, se han añadido unas imágenes para mejorar la comprensión de los datos. Estas imágenes están ubicadas en el servidor y según el valor recogido del *.xml se realiza una consulta sql a la base de datos, que devuelve la ruta de las imágenes que se mostrarán al instante en el widget.

Sube/elimina capas

Situándonos en el presente más cercano, es posible conseguir archivos .kml o .shp con datos reales que se pueden plasmar tanto en google maps como en google hearth. Tanto a nivel provincial, como nacional, se ofertan documentos con información geográfica. Por este y otros motivos está al alcance de la mano poder abastecerse de ficheros de interés para complementar lo que está hecho, y poder tenerlos junto a otros realizados a medida en un mismo entorno.

Un usuario relacionado con el mundo de la viña puede interactuar ficheros públicos ofrecidos por municipios, provincias.., como puede ser el caso de la delimitación de las parcelas del campo, que cada ayuntamiento posee en su servicio, o distribuciones de regadío, zonas sensibles.. con otros archivos realizados por profesionales, a medida del usuario, como es la situación geográfica de cada sensor y la configuración de estos.

La interfaz utilizada para este apartado de la aplicación se sitúa en la parte izquierda del entorno gráfico. No necesita mucho espacio puesto que únicamente se compone de dos formularios reducidos. El primero es el utilizado para subir archivos kml, pudiendo ser los descargados de archivos públicos, o de costo económico en otros privados, o también archivos propios de marcas realizadas por el usuario en el apartado de creación de marcas. Este formulario realiza una conexión ftp con el servidor donde se encuentra la página, y copia ahí los archivos. El registro encargado de subir el archivo al servidor aplica una capa de seguridad, contrastando que el archivo que se quiere subir es kml, y no de otro tipo como podría ser un php, donde un usuario atacante que podría haber accedido por el login, podría introducir JavaScript u otros métodos de modificación dañina.

```
<form action="subir1.php" method="post" enctype="multipart/form-data">

        <input type="hidden" value="1000000" />
        <input name="archivo" type="file" size="12"/>
        <br><br>
        <input type="submit" value="Subir Archivo" />
</form>
```

Como podemos observar, el formulario es como otro cualquiera, sólo que el tipo de datos de entrada es del tipo ‘file’ (archivo).

```php
$id_ftp = ftp_connect("ftp.parcelas.x10.mx",21);
ftp_login ($id_ftp, [redacted], [redacted]);
ftp_pasv ($id_ftp, false);
//carpeta donde vamos a deja el archivo
ftp_chdir ($id_ftp, "/public_html");

$nombreentero=explode(".",$remoto);
$archivo=$nombreentero[0];
$num=count($nombreentero)-1;
$extension=$nombreentero[$num];
if ($extension=="kml" || $extension=="KML") {

    if (ftp_put($id_ftp,$remoto,$local,FTP_BINARY)){
```

Este código corresponde a la subida de archivos. En un primer plano, se conecta al servidor por ftp, al puerto 21. Seguido se concreta la ubicación donde se van a subir los archivos. En otro plano, se trata el archivo separando sus componentes por el punto, y tratando la extensión para comprobar que es correcta. Únicamente se podrán subir archivos *.kml o *.KML, y por el usuario administrador. Con se consigue evitar que el administrador por error, o una persona de carácter maligno, intenten subir archivos php, js… con los que perturbar el correcto funcionamiento de la aplicación.

El otro sub-apartado tiene relación con las capas que el usuario desea tener ancladas en la sección home. El usuario tiene la opción de borrar alguna de estas capas que aparecen de no quitarlas mediante este medio. Se compone de una lista de capas/marcas con un checkbox a su margen izquierdo, el cual de ser clicado y mandado el formulario, realizaría la acción de suprimirlo de la lista. Para que aparezcan los elementos listados basta con hacer lo mismo que en el apartado home, una consulta a la base de datos, a la tabla de capas, e imprimirlos por pantalla. Para borrar las capas se mandan por formulario el nombre de las capas y seguido otro fichero los recoge y realiza la consulta de borrado.

En resumen, tenemos un apartado de pequeñas dimensiones y discreto, desde donde poder controlar las capas que se quieren visualizar desde home, donde estarán ancladas siempre que el usuario inicie sesión.

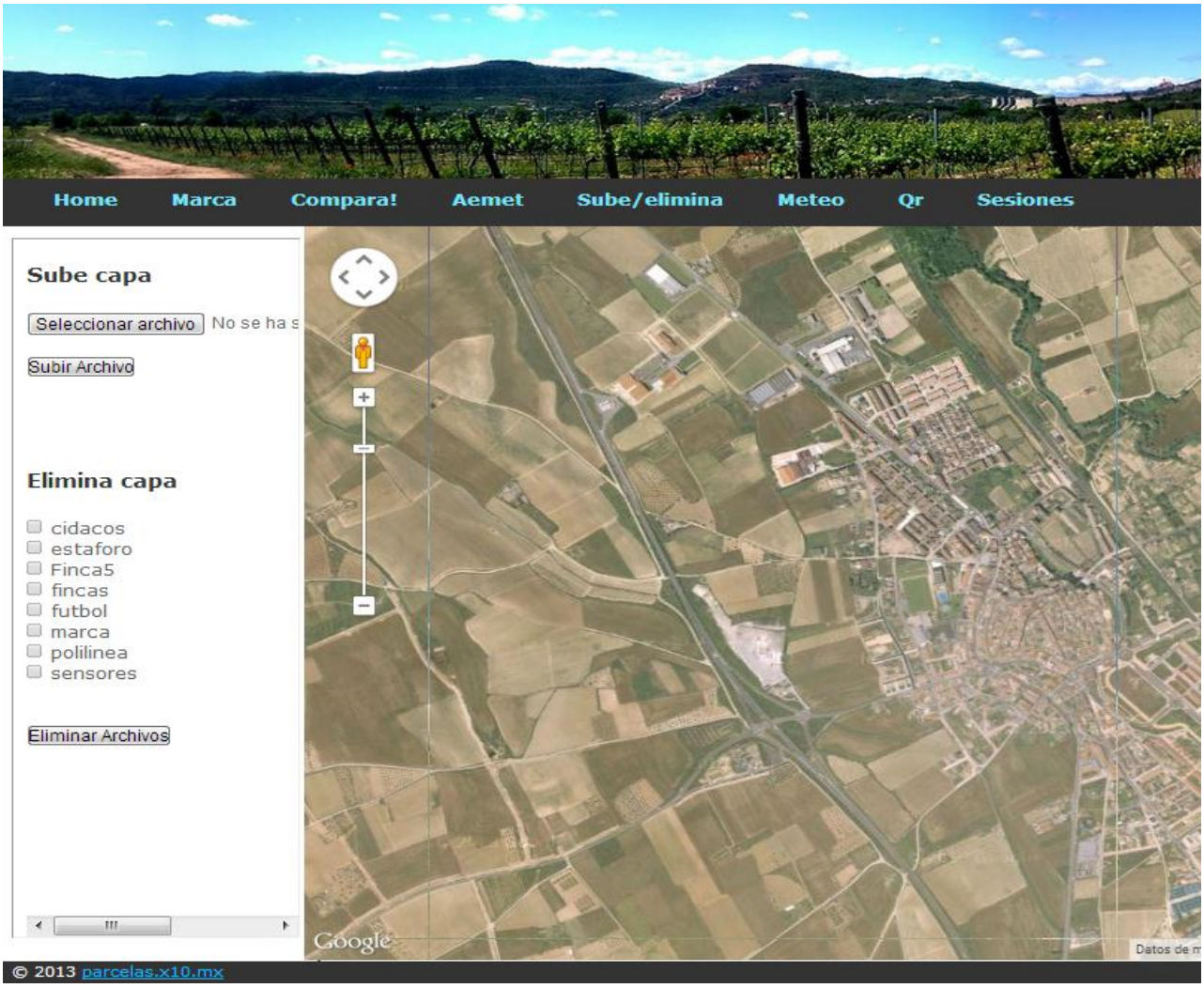


Ilustración 45: Apartado Sube/elimina

Meteo

Este apartado principalmente complementa a *Aemet*, aportando al usuario datos reales de mediciones pasadas. Con estos dos apartados se controla los datos reales del pasado, de multitud de campos, y las futuras predicciones. Si a esto le añadimos los datos recogidos por los sensores, el usuario puede comparar lo obtenido de sus sensores con lo recogido por las estaciones meteorológicas nacionales, así puede contraponerlos y comprobar que todo ha funcionado con normalidad.

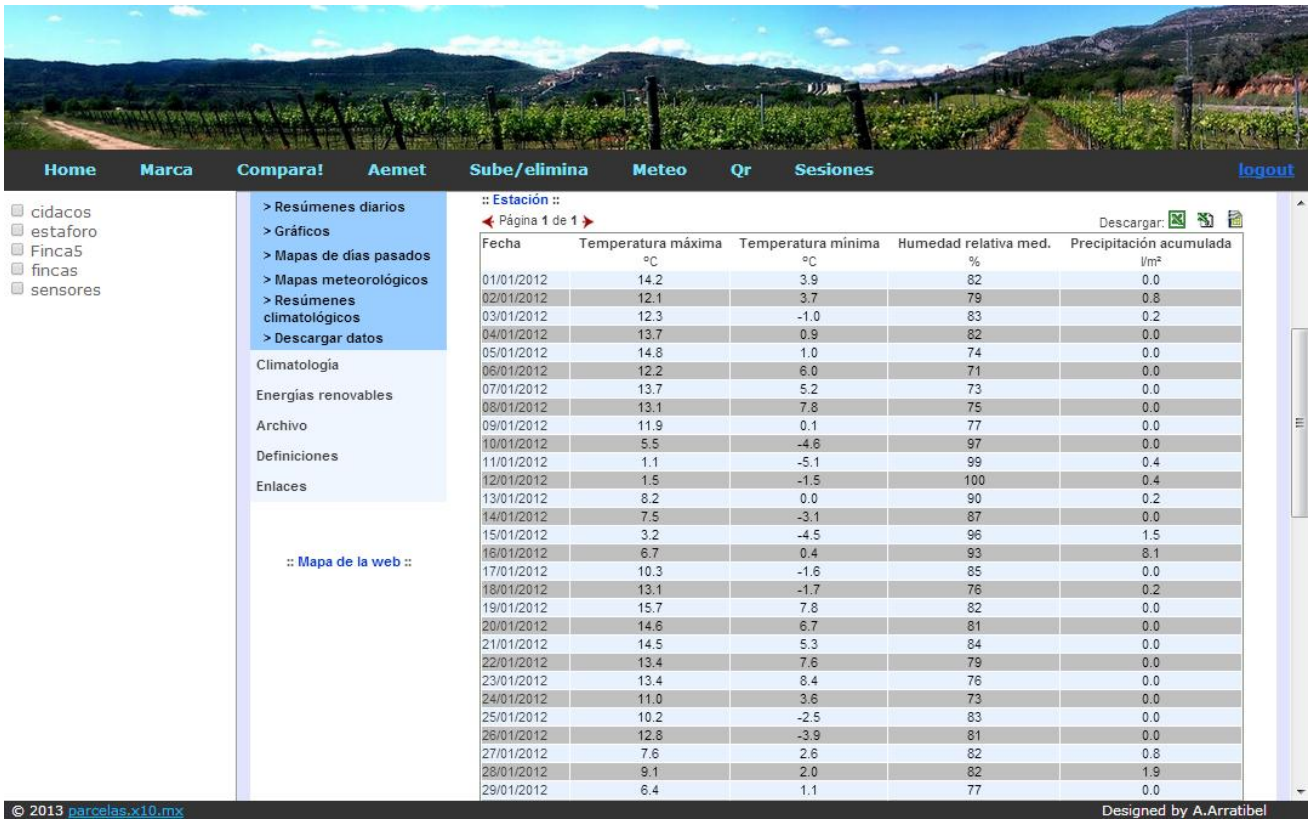


Ilustración 46: Apartado Meteo

Básicamente se realiza una visualización de una dirección url en un iframe, parecido a lo realizado en apartados anteriores, sólo que ahora en lugar de mostrar un archivo *.php se muestra una url. El lugar elegido para mostrar el iframe es el div principal de la interfaz, donde se encuentra el mapa.

En el caso de esta aplicación, los datos son proporcionados por la agencia de meteorología navarra, la cual colabora con Aemet, y ofrece datos de multitud de estaciones (automáticas y manuales), tanto para visualizarlos online como para descargarlos para Word y Excel.

Qr

Un código QR (quick response code, “código de respuesta rápida”) es un módulo útil para almacenar información en una matriz de puntos o código de barras bidimensional, esto es de relevante importancia en el sentido de que el QR puede contener mucha más cantidad de información (7.089 caracteres numéricos o 4.296 alfanuméricos). Se caracteriza por los tres cuadrados que se encuentran en las esquinas y que permiten detectar la posición del código al lector.

Hoy en día el uso del código QR se ha popularizado gracias a la combinación de dos factores:

- La publicación de las especificaciones del código. Esto ha permitido la proliferación de lectores de código QR de muy bajo coste o incluso gratuitos. Además, se han desarrollado aplicaciones de software que permiten descifrar el código QR, muchas de gratuitas y de código abierto.
- La integración con dispositivos móviles (teléfonos y PDAs). Esto ha permitido que la mayoría de los teléfonos puedan leer los códigos QR, puesto que sólo necesitan tener una cámara de fotos para la captura de los códigos y una aplicación (gratuita) para descifrar la información en los mismos. La explotación de los códigos QR utilizado como plataforma una tecnología tan madura y extendida como los smartphones, abre un abanico prácticamente infinito de oportunidades.

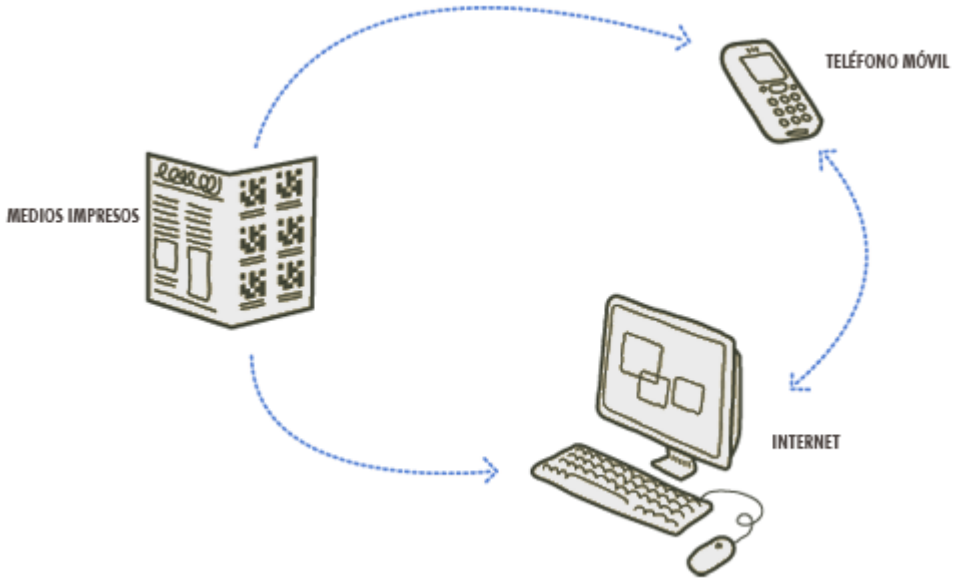


Ilustración 47: QR en el día a día

Aunque lleve en el mercado casi veinte años, la continua modernización de los teléfonos móviles, hasta lo que conocemos como smartphones, ha hecho que estos códigos no hayan sido visibles hasta hace relativamente unos pocos años. En el día a día nos encontramos con multitud de estos códigos, por las calles, internet, prensa… por ello este apartado hace que la aplicación sea verdaderamente actual.

Esta sección se ha creado principalmente para facilitar la comunicación entre los usuarios de la aplicación. El usuario podrá crear códigos con varios fines: texto, email, teléfono, geolocalización… Las marcas que se pueden visualizar en la aplicación pueden ser trasladadas a código Qr, para así mejorar la velocidad de actuación, y poder interactuar con las tecnologías que están a nuestro alcance. Además, el usuario podrá hacer anotaciones, emails o sms's, guardarlos y automáticamente se pasarán a dispositivos propios provistos de lectores de códigos Qr.

La implementación de este apartado es relativamente sencilla. Todo el apartado visual se sintetiza en un gran formulario, dentro de él lo primero que se encuentra son las diferentes opciones de QR a producir, dependiendo de cuál se elija, JavaScript desplegará los apartados que le corresponden. Una vez rellenados los datos, una librería de JavaScript, jQuery, se encarga de transformarlos en un código Qr descargable.

Ilustración 48: Apartado QR

Sesiones

Ahora pasamos a ver la última funcionalidad de la aplicación, que sólo estará disponible para el usuario administrador. Esta aplicación está diseñada principalmente para que una o dos personas la gestionen, controlando los datos y realizando las marcas y anotaciones oportunas, pero también está bien que una persona relacionada en un momento con el entorno pueda tener acceso a la interfaz. Por esto, el usuario administrador tendrá el poder de crear cuentas a nivel de usuario, esto quiere decir que podrá disponer de todas las funcionalidades de la aplicación, a excepción de aquellas que puedan entrañar riesgos para la aplicación, como la subida de archivos o la gestión de sesiones.

La estructura de este apartado se fundamenta en dos formularios. El primero se utiliza para dar de alta nuevos usuarios, sólo se necesita rellenar los campos de 'nombre' y 'pass', y seguido, al dar de alta al usuario se realizará una inserción en la tabla de usuarios de la base de datos, cifrando la contraseña con md5. El segundo formulario lista todos los usuarios a excepción del administrador, y a continuación se pueden elegir aquellos que dar de baja, y al enviar el formulario se realizará una consulta sql de DELETE.

Capítulo 4 Conclusiones y Líneas futuras

En este apartado se exponen las conclusiones que se han obtenido del trabajo realizado, valorando los objetivos cumplidos que se plantearon al principio del proyecto. También se analizan los posibles trabajos futuros que se podrían llevar a cabo para mejorar este proyecto. En último lugar se mostrarán las referencias bibliográficas utilizadas para poder realizar este proyecto.

4.1. Conclusiones

El proyecto comenzó con la idea de mostrar una red de sensores en una aplicación, primeramente se pensó en anclar dichos sensores a sus respectivas parcelas en el mapa, introduciendo coordenadas y código en el mismo código HTML/PHP. Esto reducía la interacción con el usuario de forma notable. Posteriormente, se hizo un estudio sobre los ficheros SHP y KML, y la relación de estos con Google Maps. De esta forma se conseguía tener por separado el código de la página del código de los sensores y parcelas.

Se ha realizado un trabajo intenso en cuanto al estudio de documentación acerca de la API de Google y la manera de conseguir alcanzar los objetivos propuestos mediante esta herramienta gratuita. Con este trabajo nos podemos hacer una idea del gran número de posibilidades que existen para tratar los mapas e incluirlos en la aplicación, pero también algunas limitaciones inherentes a la propia herramienta que no permitían hacer lo deseado en el proyecto.

La principal limitación fue la relación entre el lenguaje JavaScript y el lenguaje PHP. Con el primero se trataba la representación de gráficas, mapas, calendarios… Situarlos en la interfaz de la página de la forma deseada fue algo costoso, pues había que pasar las variables de un lenguaje a otro, todo ello en la misma página.

Una vez resuelto este inconveniente, la siguiente fase era hacer que el resto de funcionalidades fueran de acceso rápido. Para ello se optó por la fluidez que ofrecen los *iframes* en este campo, evitando tener que recargar toda la página para acceder a una funcionalidad, teniendo un total de ocho. Este modelo también trajo consigo problemas, para comunicar el php con el *iframe*, pero se realizaron archivos auxiliares para interrelacionarlos. Con el resultado final, la página y sus distintas funcionalidades son accesibles y ejecutables en la misma pestaña del navegador, sin tener que modificar la url, de una forma totalmente dinámica.

Todos estos inconvenientes encontrados por el camino han sido resueltos de manera positiva y satisfactoria, puesto que se han cumplido los objetivos propuestos al inicio del proyecto además del valor añadido que supone haber aprendido a usar tecnologías con las que no había trabajado y que actualmente tienen mucha relevancia como son Ajax, JavaScript, librerías PHP.

4.2. Líneas futuras

Este proyecto es ampliable en muchos aspectos, con más tiempo y trabajo podría ser la aplicación que muchas empresas querrían tener para desarrollar parte importante de su trabajo. El primer aspecto a mejorar creo que sería la relación entre usuarios y parcelas. Ahora mismo con este proyecto todas las parcelas son gestionadas por el administrador, y el resto de usuarios creados por él pueden gestionar sólo las parcelas que él haya introducido en el sistema. En futuros proyectos, estaría bien que cada usuario gestionara sus propias parcelas y sensores, relacionando ambos campos en la base de datos, así cuando un usuario accediera a la aplicación sólo vería las parcelas que él mismo ha configurado.

Además de recoger un tipo de valor cada sensor, sería deseable que pudiera abarcar más opciones. Esto se complementaría con una toma de decisiones a la hora de tomar las medidas; ahora mismo los sensores enviarían datos cada cierto tiempo, según quiera el administrador, pero en una futura mejora el usuario podría ordenar al sensor de forma manual y a distancia desde su propio pc, *tablet* o s*martphone*, cuándo desea tomar la medida, si en ese mismo momento o programar una medida cuando él lo desee.

Con las ideas anteriores se mejoraría el concepto de este proyecto, pero este proyecto es aplicable a otras áreas. Un ejemplo podría ser la distribución de sensores en un edificio o empresa industrial. En ese caso, se monitorizarían los datos de los sensores para medir calor, humedad, humos, ruido, sol… y controlar estos valores para realizar un consumo responsable de calefacción, aire acondicionado, sistemas de extracción de humos… Estos campos son semejantes a los que actualmente se controlan en las *smart cities,* en pos de un desarrollo sostenible y un uso eficiente de los recursos de los que disponemos.

4.3. Referencias bibliográficas

- "*Sistemas de Información Medioambiental*" [2005]
  Autores: José A. Taboada González y José Manuel Cotos Yáñez
- Monitorización medioambiental: http://www.balmart.es/
- API Google: https://developers.google.com
- Documentación KML: https://developers.google.com/kml
- Google Maps: https://developers.google.com/maps
- PHP: http://www.php.net
- Funcionalidades PHP: http://forums.phpfreaks.com
- Tutoriales/Ejemplos código: http://www.w3schools.com
- Wikipedia: http://es.wikipedia.org
- IDENA, Infraestructura de Datos Espaciales de Navarra: http://idena.navarra.es
- Descargas SHP/KML: http://idena.navarra.es/descargas
- Herramienta SHP/KML: http://www.qgis.org
- IDECanarias, Infraestructura de Datos Espaciales de Canarias: http://www.idecan.grafcan.es
- AEMET: http://www.aemet.es
- Meteo Navarra: http://meteo.navarra.es
- Congreso Nacional Medio Ambiente: http://www.conama10.es
- PHPExcel: http://phpexcel.codeplex.com
- Gráficas JavaScript: http://www.amcharts.com
- Colores: http://jscolor.com
- Herramientas: http://www.mapmash.in http://www.deperu.com/generador-qr/#wifi
- Arquitectura en 3 capas: http://www.slideshare.net/Decimo/arquitectura-3-capas

Capítulo 5 Anexo

5.1. Manual de usuario

En esta sección de los anexos vamos a proceder a explicar algunos funcionamientos de las distintas partes de la interfaz web.

Visualización de capas

Lo primero que el usuario visualiza tras pasar la sección de autentificación es el apartado principal *Home* y el mapa de interacción. A su izquierda se encuentran las distintas capas que el usuario puede seleccionar, seleccionamos las fincas y los sensores clicando en los *checkbox* correspondientes, quedando un resultado así

Ilustración 49: Visualización de capas

Seguidamente, el usuario puede proceder a clicar en un sensor, y le aparecerán las distintas opciones que tiene ese sensor

Ilustración 50: Opciones de sensor

Cada una de las opciones corresponde con una de las gráficas de representación de los datos, clicando sobre ellas se abrirá una nueva pestaña con la gráfica

En caso de la gráfica de barras, el usuario puede introducir las fechas entre las que desea los datos, bien manualmente o mediante el calendario desplegable. El color también es configurable y requiere que se clique sobre él para que se despliéguela paleta. Una vez ajustado todo, aparecerá un mensaje para confirmar las fechas y seguido mostrará los nuevos datos.

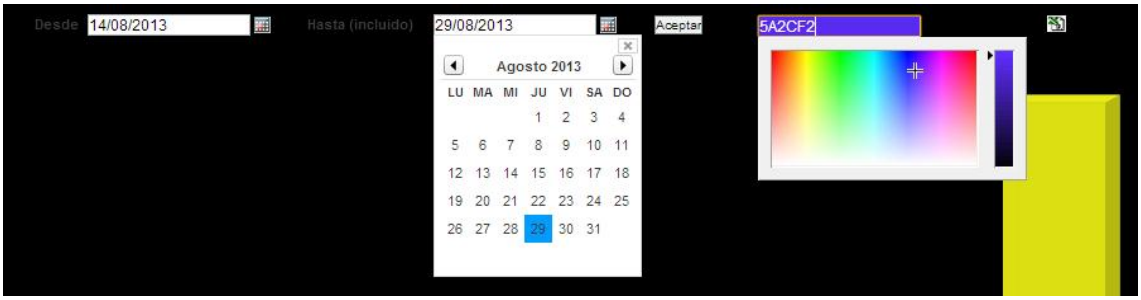


Ilustración 51: Menú gráfica sensor

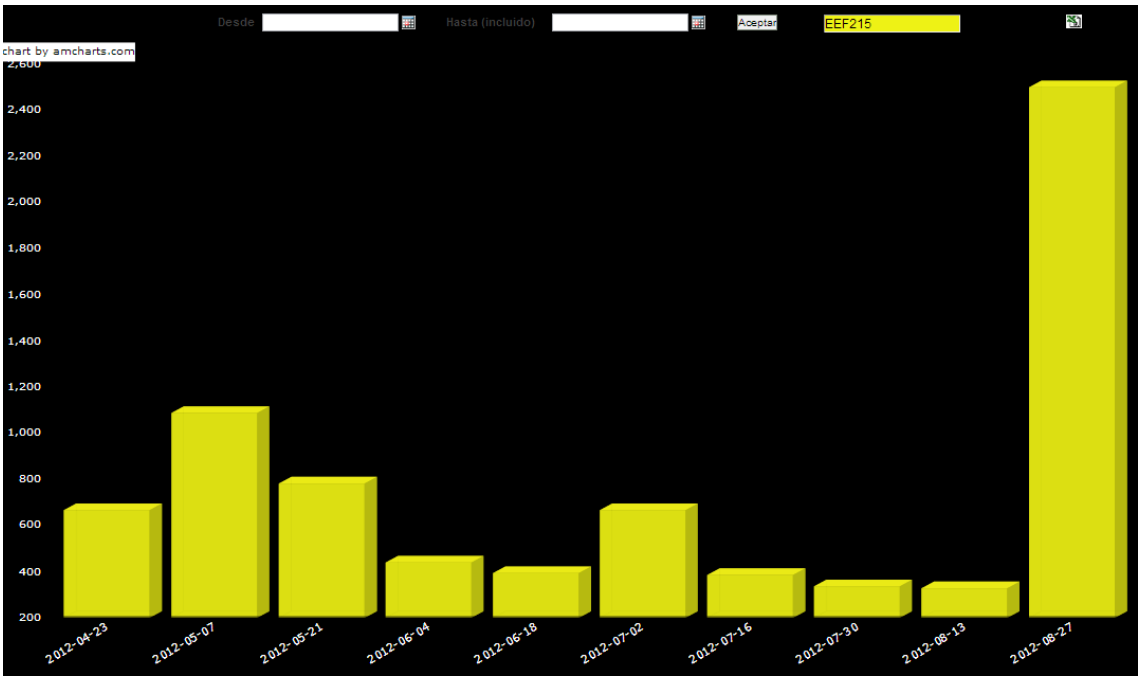


Ilustración 52: Gráfica de barras

Si por el contrario se elige la opción “sectores”, aparecerá una gráfica con formato parecido y la gráfica como esta

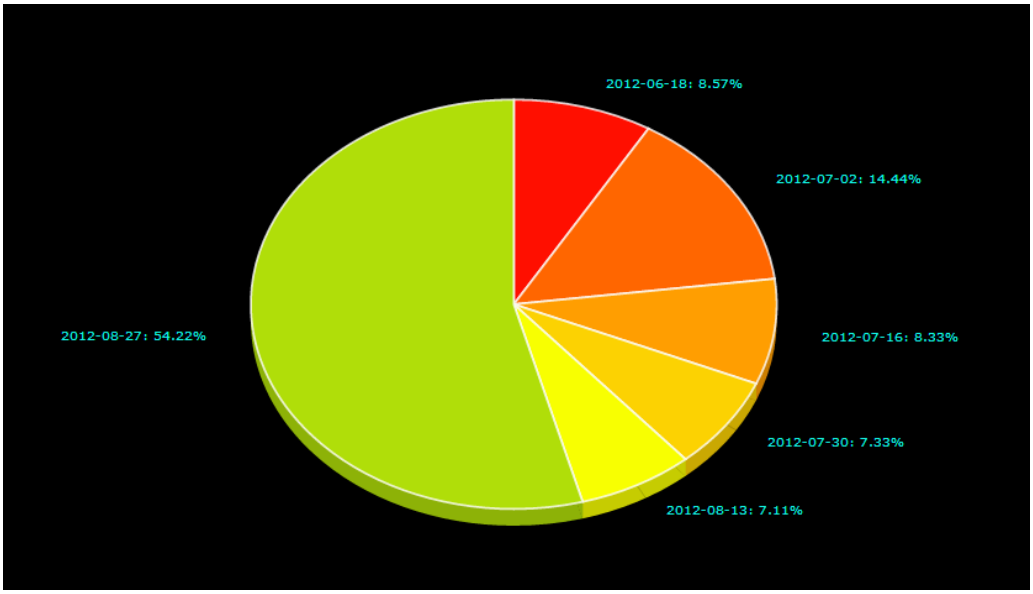


Ilustración 53: Gráfica de sectores

Si el usuario clica sobre un “quesito”, este se saldrá del sector de esta manera, diferenciándolo del resto

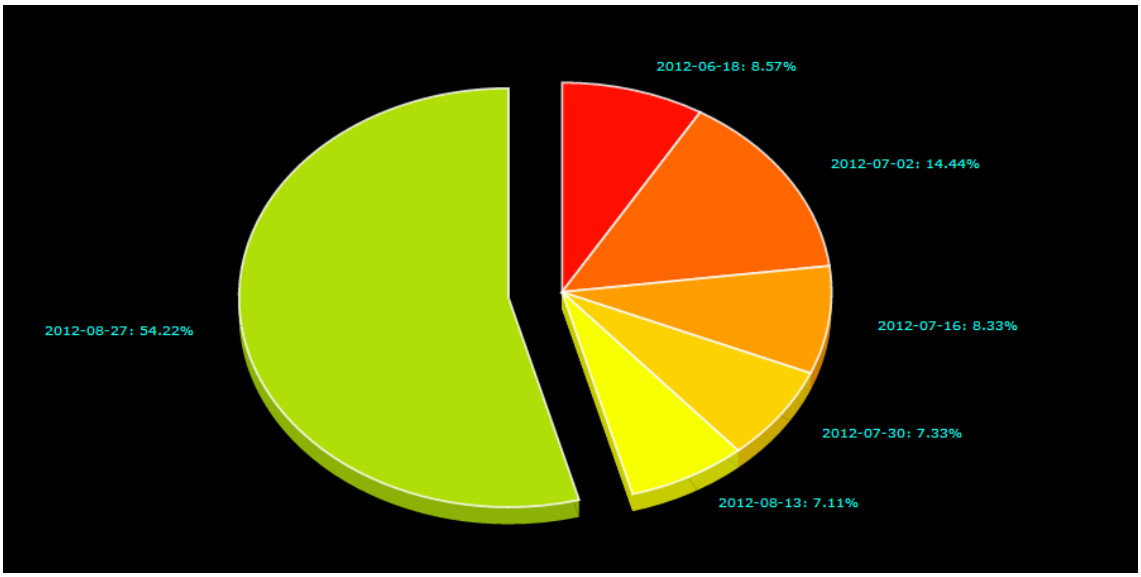


Ilustración 54: Separación de sectores

Si se desean descargar los datos sólo habrá que clicar en el icono Excel en el margen superior derecho.

Creación y subida de marca

A continuación vamos a crear un polígono, en el mapa, correspondería a una parcela o un área delimitada. Para ello vamos al apartado *Marca* y seleccionamos la opción de polígono

Ilustración 55: Opciones de marca

Seguidamente tendremos que dibujar el polígono en el mapa que se aparece para ello, de esta forma

Ilustración 56: Creación de polígono

Si el usuario está conforme, puede configurar las opciones que aparecen y a continuación darle a guardar. Si no está conforme o quiere cambiar algún punto lo puede hacer con la etiqueta *Deshacer*. Más tarde, necesitamos el código KML para crear el fichero, para ello seleccionamos *Guardar como KML*, y seguido *Generar KML* y aparecerá el código correspondiente seleccionado para que el usuario lo copie y lo pegue en un archivo con extensión *.kml

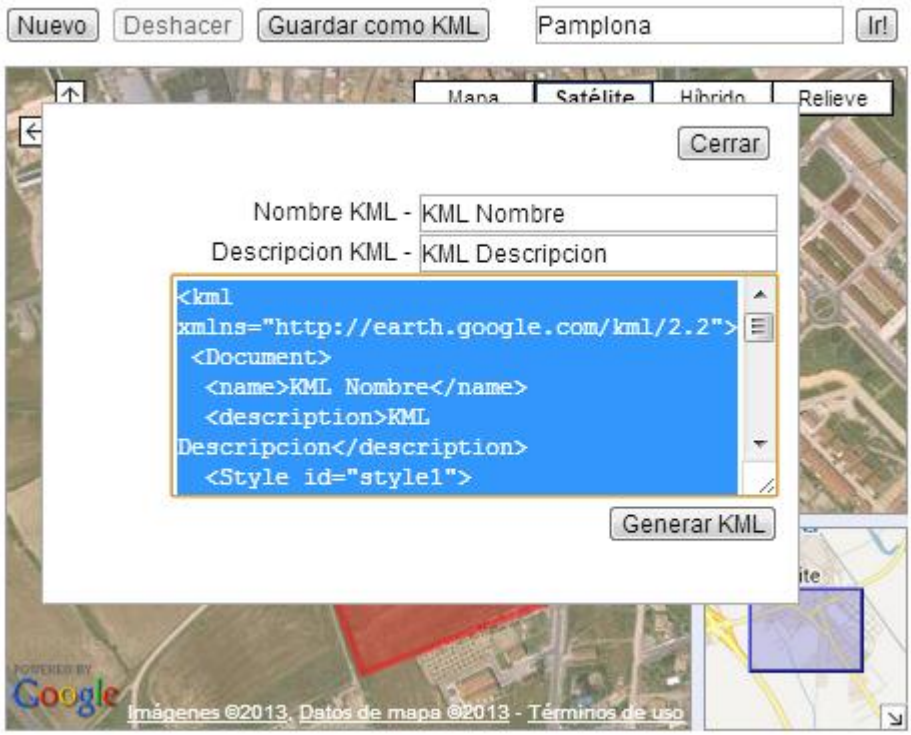


Ilustración 57: Código KML de polígono

El nuevo archivo creado lo he llamado “Poligono.kml” y en él he pegado el código generado anteriormente. Ahora nos dirigimos a la sección *Sube/elimina* , seleccionamos el archivo y le damos a “Subir archivo”. Si está bien subido aparecerá un mensaje así

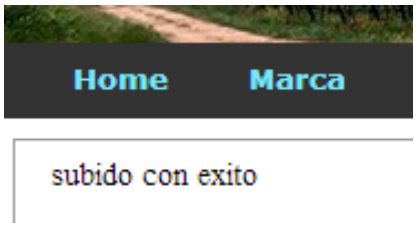


Ilustración 58: Archivo subido

Finalmente, de vuelta en el apartado *Home*, tendremos nuestro archivo listo para ser visualizado en el mapa

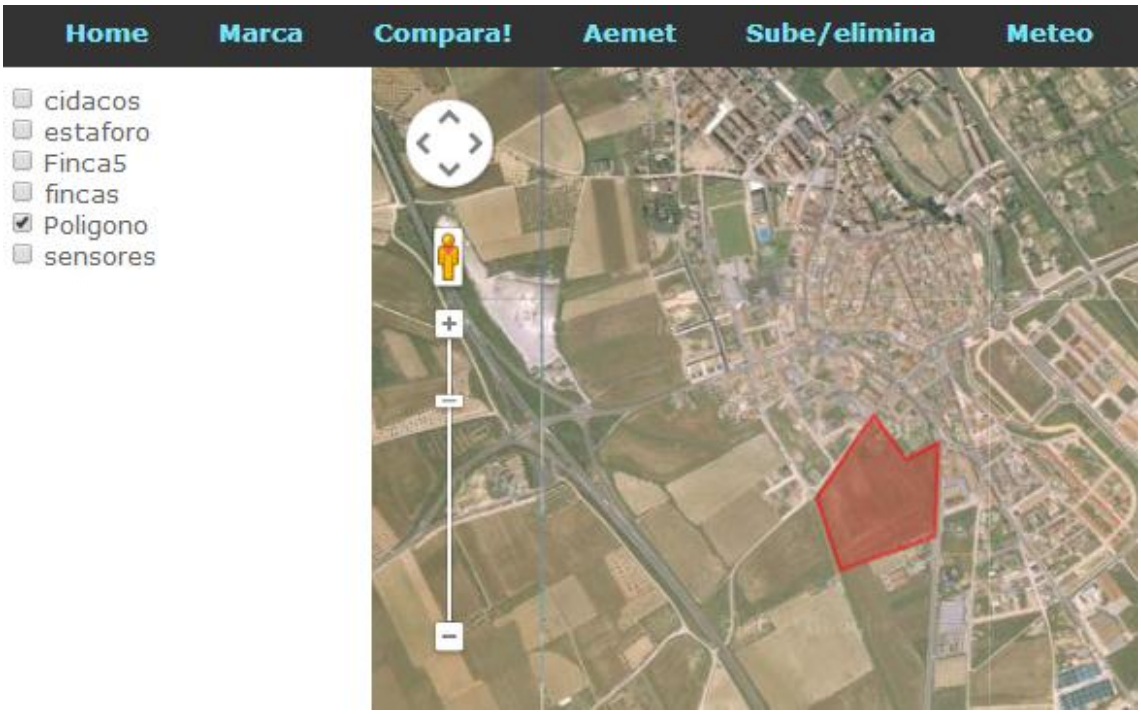


Ilustración 59: Visualización de archivo

Como sólo ha sido un prueba y no nos interesa esta marca, vamos de nuevo a la sección *Sube/elimina*, seleccionamos la marca y afirmamos que se elimine

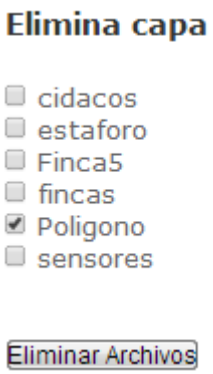


Ilustración 60: Archivos a eliminar

Gráfica compara

La interfaz de esta gráfica con respecto a la de sectores anterior es casi igual, a excepción de la leyenda inferior. En un primer momento tenemos los datos de todos los sensores

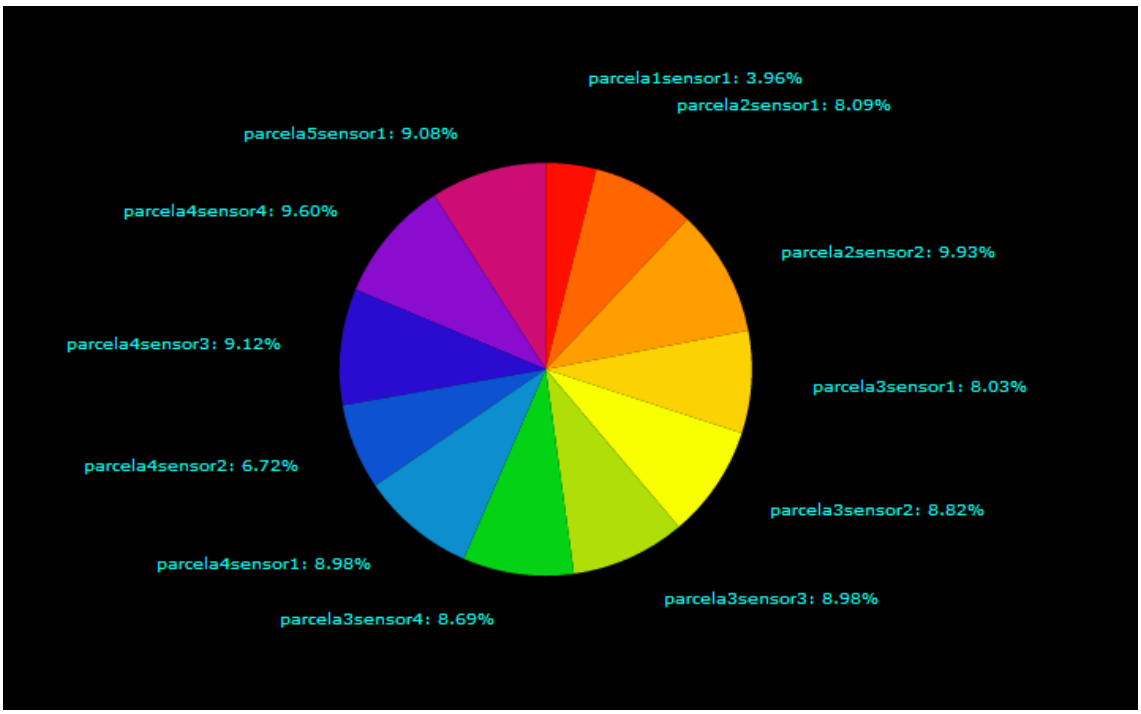


Ilustración 61: Gráfica compara

Si queremos quitar los datos, por ejemplo del sensor "parcela4sensor3", de tono azul oscuro, elegimos este sensor en la parte de abajo y clicamos de este modo

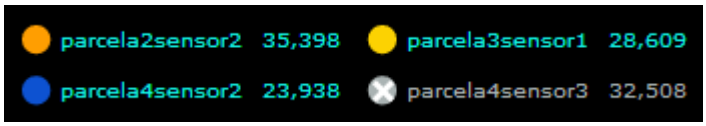


Ilustración 62: Quitar parcela

Se quedará en un todo gris esta selección y la gráfica cogerá los valores de los nodos restantes

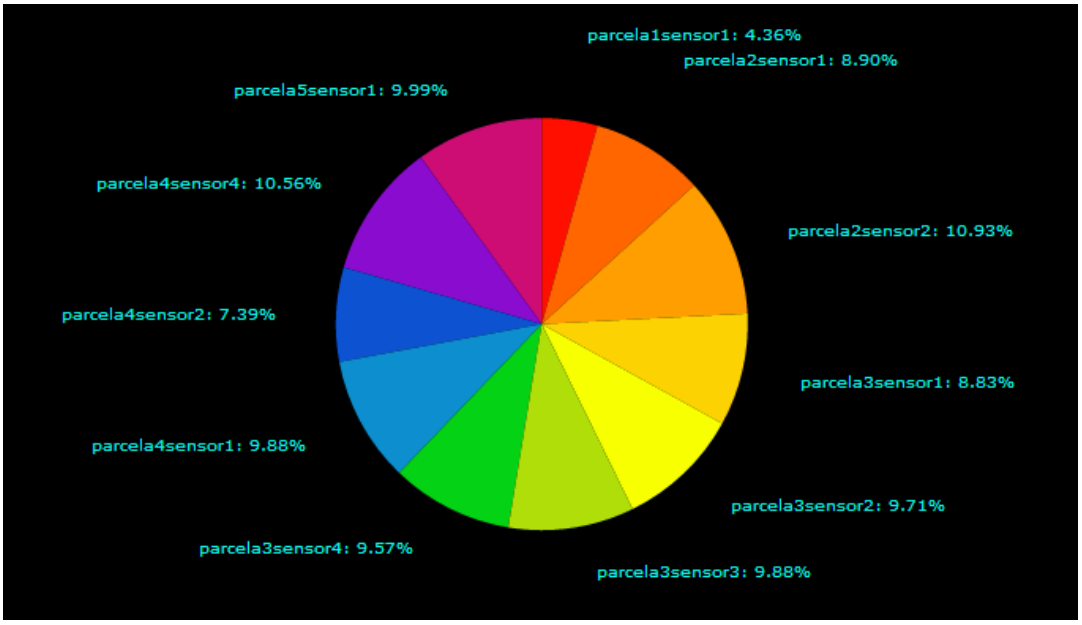


Ilustración 63: Gráfica compara sin un sensor

A continuación, podemos sacar los "quesitos" como hemos hecho antes, y también introducir las etiquetas dentro del circulo con este resultado

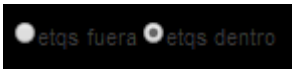


Ilustración 64: Opciones etiquetas

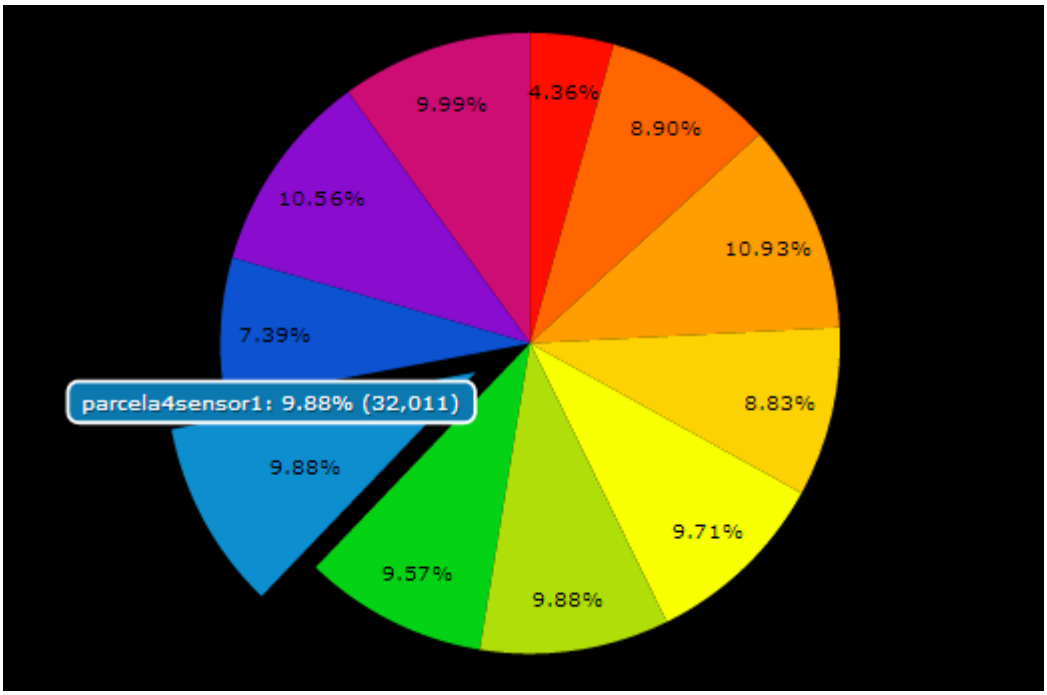


Ilustración 65: Etiquetas dentro

Por último, si queremos conseguir un aspecto 3D como en la gráfica de sectores del apartado anterior, basta con elegir 3D en el menú inferior

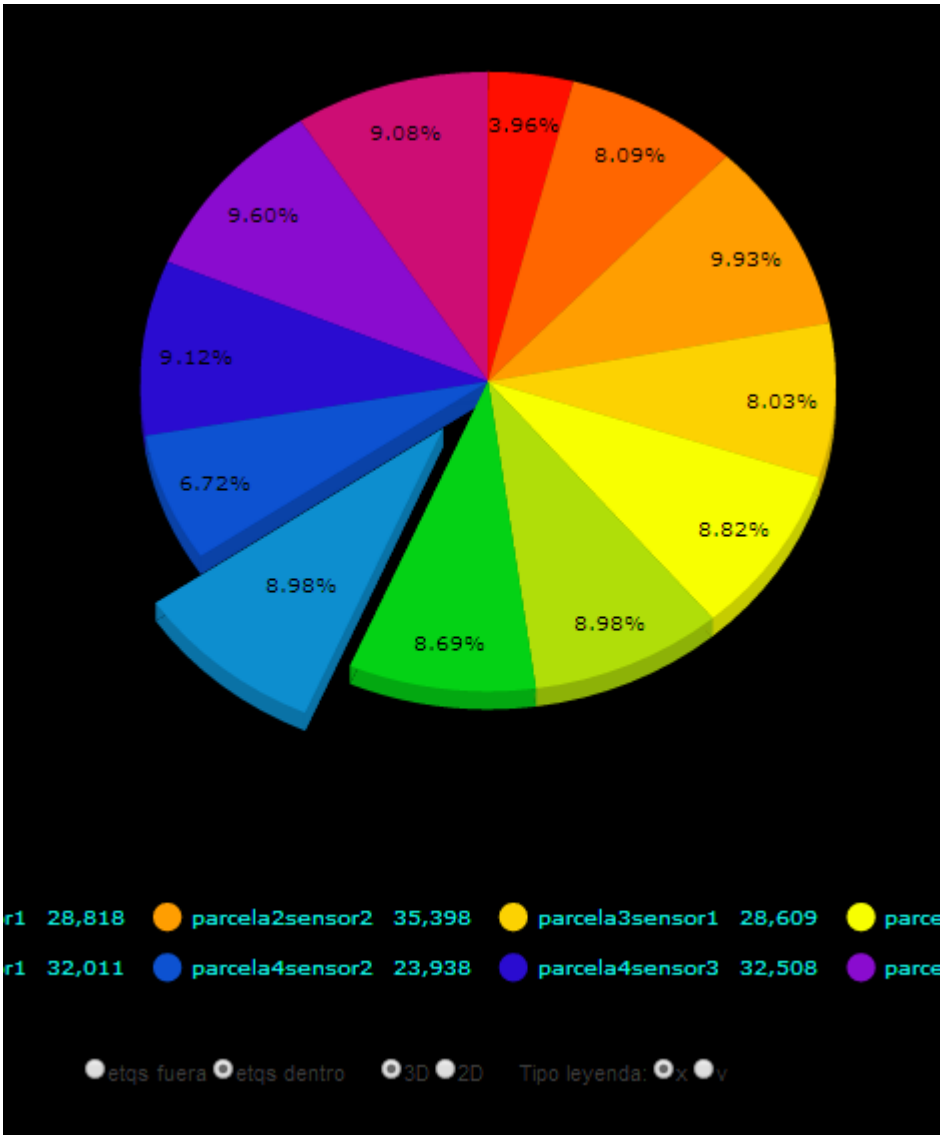


Ilustración 66: Gráfica compara 3D

Crear código QR

Para la creación de un código Qr basta con ir a la sección *Qr* y seleccionar el tipo de Qr que se desee, por ejemplo una ubicación, a continuación podemos introducir la dirección a buscar, como hacemos en google maps o introducir directamente latitud y longitud. Una vez rellenado el formulario, clicaremos en *Generar código QR,* y automáticamente aparecerá, con un enlace debajo por si el usuario desea descargarlo al terminal. El resultado sería este

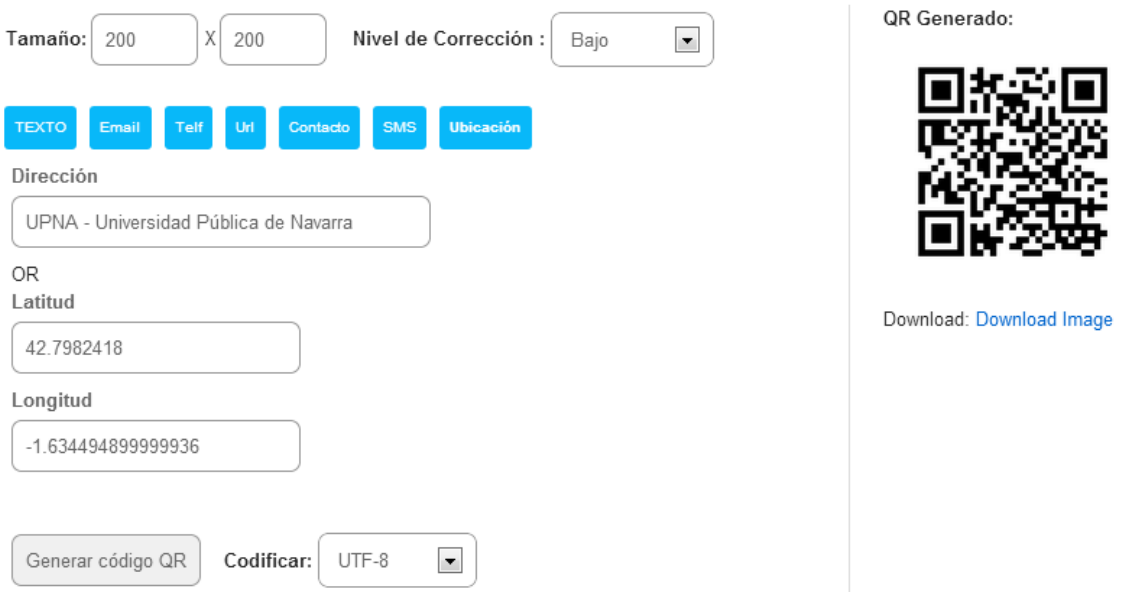


Ilustración 67: Creación código QR

Alta de usuario

Para dar de alta un usuario simplemente vamos a la sección *Sesiones* y rellenamos el nombre y contraseña del nuevo usuario

Ilustración 68: Alta usuario

Si el sistema tiene éxito en la inserción del nuevo usuario, aparecerá este mensaje

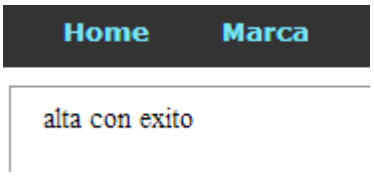


Ilustración 69: Mensaje de alta

Y seguido ya estará disponible esta cuenta para ser eliminada por el administrador

Ilustración 70: Elimina sesión

Salir de la aplicación

Para abandonar la aplicación, es aconsejable que el usuario haga uso del botón *logout*, puesto que si cierra el navegador, la sesión seguirá estando activa, como ocurre con el correo electrónico u otras aplicaciones

Ilustración 71: Logout